

### Laden einer Bibliothek

#### Importanweisung

`import modulname`

`import modul as short`

`from modul import *`

#### Befehlsaufruf

`modulname.befehl()`

`short.befehl()`

`befehl()`

`dir(modulname)` zeigt alle Befehle im Modul an

`help(befehl)` zeigt die Hilfe für den Befehl an

### Module

`random`

Zufallszahlen

`math`

Mathematik

`turtle`

Turtle-Graphik

`os`

Betriebssystem

`sys`

Console

### Zufallszahlen

`randint(a,b)`

Zufallszahl  $a \leq x \leq b$

`randrange(a,b)`

Zufallszahl  $a < x < b$

`randrange(b)`

Zufallszahl  $0 < x < b$

`random()`

Zufallszahl 0.0 - 1.0

### Turtlegraphik

`fd(s)`

Vorwärts s Einheiten

`bk(s)`

Rückwärts s Einheiten

`rt(a)`

Drehen nach Rechts

`lt(a)`

Drehen nach Links

`pu()`

nicht Zeichnen

`pd()`

ab jetzt Zeichnen

`circle(r)`

Kreis mit Radius r

`fillcolor(f)`

Füllfarbe setzen

`begin_fill()`

ab jetzt Ausfüllen

`end_fill()`

Ausfüllen beenden

`home()`

zum Startpunkt

`reset()`

neu Anfangen

s: Anzahl Pixel

a: Winkel in Grad

f: Farbe z.Bsp. 'red'

### Ein- und Ausgabe

`var = int(input('prompt'))`

`var = float(input('prompt'))`

`print('prompt')`

`print('text %s text' %(var))`

`print('text {} text'.format(var))`

%s Platzhalter Textvariable, %d Platzhalter Zahlvariable

### Operatoren

`x+y`

Addition

`x-y`

Subtraktion

`x*y`

Multiplikation

`x/y`

Division

`x%y`

Modulo

`x**y`

$x^y$

`x//y`

Division ohne Rest

### Datentypen

Integer

-25, 23

Float

-2.34, 65.3

String

'Hello', "World", ""multiline""

Boolean

True, False

List

[value, ...]

Tupel

(value, ...)<sup>1</sup>

Dictionary

{key:value,...}

Set

{value, value,...}<sup>2</sup>

<sup>1</sup> Klammern optional

<sup>2</sup>set() erzeugt eine leere Menge

### Funktionen

`def funktionsname(Var1, Var2=4):`

`#Anweisungen`

`#Anweisungen`

`return result #optional`

### Selektionen

`if bedingung:`

`#Anweisungen, falls bedingung erfüllt ist`

`elif bedingung2:`

`#Anweisungen`

`else:`

`#Anweisungen`



### Bedingungen

<	kleiner als	a < 10
>	größer als	b>4
==	gleich	c=='yes'
<=	kleiner gleich	d<=5
>=	größer gleich	e<=7
!=	ungleich	g!='no'
'in'	in	'x' in 'mexico'
'not in'	nicht in	y not in 'mexico'

### Zeichenketten (Strings)

str.lower()	in Kleinbuchstaben umwandeln
str.upper()	in Grossbuchstaben umwandeln
str.replace(old,new)	old durch new ersetzen
str.split()	Teilt den String auf
str[1:5]	Zeichen 1-5 anzeigen
list(str)	erzeugt eine Buchstabenliste

### Strings Slicing

0	1	2	3	4	5	6	7	8	9	10	11
M	o	n	t	y		P	y	t	h	o	n
-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

### Iterationen

#### for-Schleifen

```
for item in list:
    #Anweisungen für item
    #Anweisungen für item'
#Anweisungen nach der Schleife
for i in range(n):
    #Anweisungen n mal Wiederholen
```

#### while Schleife

```
while bedingung:
    #Anweisungen
```

range(n) = [0,1,2,3,...,n-1] Liste mit den ersten n Zahlen  
break beendet die Schleife. continue beendet den aktuellen Durchlauf

### Arbeiten mit Listen

len(myList)	Länge von myList
myList[i]	i-tes Element der Liste
myList[i:j]	Ausschnitt von i bis j
x in myList	True wenn x in myList ist
myList.append(x)	x myList anhängen
myList.insert(i,x)	x vor der Stelle i einfügen
myList[i]=x	Element i ersetzen
myList.remove(x)	entfernt x aus myList
myList.pop([i])	entfernt das i-te Element
myList=[]	

### Dictionarys

len(dict)	Länge von dict
del dict[key]	löscht den Schlüssel key
dict.keys()	Liste von Schlüsseln
key in dict	Wahr wenn es den Schlüssel gibt
dict = {key: value, }	

### Exception Handling

```
try:
    #Anweisungen
except exception type as var:
    #Anweisungen
finally:
    #Anweisungen
```

### Dateien

open(file,mode)	Datei öffnen
f.read()	liest den ganzen File
f.readline()	liest eine Zeile
f.readlines()	liest alle Zeilen
for line in f:	Zeile für Zeile durchgehen
f.write(prompt)	schreibt in die Datei
f.close()	schliesst die Datei
mode: 'r' lesen, 'w' schreiben,'r+' lesen und schreiben, 'a' anhängen readlines() erzeugt eine Liste von Zeilen	

