Generell

Die numerischen Datentypen sind gleichermaßen zu behandeln wie in den bekannten Programmiersprachen.

Datentypen

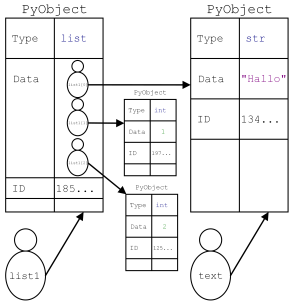
Datentyp	Beschreibung	False-Wert
NoneType	Indikator für nichts	None
Numerische Datentypen		
int	Ganze Zahlen	0
float	Gleitkommazahlen	0.0
bool	Boolesche Werte	False
complex	Komplexe Zahlen	0 + 0j
Sequenzielle Datentypen		
str	Zeichenketten oder Strings(unveränderlich)	’ ’
list	Listen(veränderlich)	[]
tuple	Tupel(unveränderlich)	()
bytes	Sequenz von Bytes(unveränderlich)	b’ ’
bytesarray	Sequenz von Bytes(veränderlich)	bytearray(b’ ’)
Mengen		
set	Einmalig vorkommende Objekte	set()
frozenset	Wie set jedoch unveränderlich	frozenset()
Assoziative Datentypen		
dict	Dictionary(veränderlich)	

Operatoren

Operator	Beschreibung
x // y	Ganzzahliger Quotient
x ** y	Potenzieren, x^y
+,-,...	Übliche Operation

Variablen

```
list1 = [text, 1, 2]
```



Casting

Datentyp	Klasse	Direkt	Datentyp	Klasse	Direkt
Ganzzahl	int()	3	Gleitkommazahl	float()	3.1415
Boolescher Wert	bool()	True,False	Komplexe Zahl	complex()	2 + 4j
String	str()	”HSR”,’OST’	Liste	list()	[1,2,3]
Menge	set()	1,2,3	Tupel	tuple()	(1,2,3)
Unver. Menge	frozenset()	frozenset(1,2,3)	Dictionary	dict()	, ”Key”: 1

Eingabe und Ausgabe

```
name = input("type your name:")
print("Hello", name)
strList = ["YES", "WE", "CAN"]
for w in strList:
    print(w, end="--")
# type your name: MFG GG OG
# Hello MFG GG OG
# YES--WE--CAN--
```

Parameter	Beschreibung	Default
object(s)	Alle Objekte werden in String konvertiert	
sep=’seperator’	Separierung der Objekte	’ ’
end=’end’	Letztes Zeichen des print-Befehl	’\n’
file	Objekt mit einer Write-Methode	sys.stdout
flush	Boolscher Wert für die Output-Überprüfung	False

Listen

```
l1 = [1, 4, 3, 2]          # Listen erstellen
l2 = ["Panda", "Po"]
l2.append("Fu")            # l2 = ['Panda', 'Po', 'Fu']
l1 = l1 + [5]              # l1 = [1, 4, 3, 2, 5]
l1.extend(l2)              # l1 = [1, 4, 3, 2, 5, 'Panda', 'Po', 'Fu']
l1.remove("Po")            # l1 = [1, 4, 3, 2, 5, 'Panda', 'Fu']
l1.insert(5, 6)            # l1 = [1, 4, 3, 2, 5, 6, 'Panda', 'Fu']
l3 = l1[0:6:1]             # l3 = [1, 4, 3, 2, 5, 6] l1[0] exklusive l1[6]
l3.sort(reverse=False)     # l3 = [1, 2, 3, 4, 5, 6]
l4 = l3[::2]               # l4 = [1, 3, 5] ; jedes zweite Element auswaehlen
```

Tupel