# Datentypen in Python

Python kennt zwei Arten von Typen: **primitive Typen** und **Objekttypen.**

* Der Wert eines primitiven Typs wird direkt gespeichert. Bei einer Zuweisung an eine andere Variable wird der Wert kopiert.
* In Objekttypen wird nicht der Wert, sondern ein Link (Referenz) auf das Objekt gespeichert. Bei einer Zuweisung an eine andere Variable wird nur die Referenz kopiert, nicht das Objekt. Objekttypen werden mit der Referenz None initialisiert.

## Primitive Typen

|  |  |  |
| --- | --- | --- |
| **Typname** | **Beschreibung** | **Bereich / Beispiel** |
| **Ganze Zahlen** | | |
| **int** | ganze Zahl in 32 Bit | -2147483648 … 2147483647  a = 12 |
| **long** | ganze Zahl mit mehr als 32 Bit | a = 12L |
| **Gleitkommazahlen** | | |
| **float** | einfache Gleitkommazahl (32 Bit) | b = 0.5  c = 2.6e+7  d=0.1E-5 |
| **Wahrheitswerte** | | |
| **boolean** | boolscher Wert (1 Bit) | True (1)  False (0) |

## Zeichenketten und Listen als Objekttypen

|  |  |  |
| --- | --- | --- |
| **Typname** | **Beschreibung** | **Bereich / Beispiel** |
| **Zeichenketten** | | |
| String | Zeichenkette beliebiger Länge  Zugriff auf das erste Zeichen  Zugriff auf das i-te Zeichen  Zugriff auf das letzte Zeichen | s = “Zeichenkette“  s[0]  s[i]  s[-1] |
|  | Methoden:  Länge der Zeichenkette  Suche einer Teilkette (liefert die Position >=0 oder -1  Umwandlung in Groß-/Kleinbuchstaben | len(s)  s.find(“eiche“)  s.find(“a“)  s.upper()  s.lower() |
| **Listen** | | |
| Liste | Geordnete Folge von Elementen beliebiger Länge  Zugriff auf das erste Element  Zugriff auf das i-te Element  Zugriff auf das letzte Element  Teilliste mit Index i<= x <j  Liste klonen | liste = [1,3,2,5,6]  liste[0]  liste[i]  liste[-1]  liste[i:j]  liste2 = liste[:] |
| boolean | Methoden:  Länge der Liste  Element anhängen  Element entfernen  Element mit dem Index i lesen und entfernen  Liste sortieren  Liste umdrehen | **len(liste)**  **liste.append(elem)**  **liste.remove(wert)**  **liste.pop(i)**  **liste.sort()**  **liste.reverse()** |

# Typumwandlungen Python

Bestimmung des Typs der Variablen var: type(var)

Umwandlung der Variablen var in einen String: str(var)

Umwandlung der Variablen var in Integer: int(var)

Umwandlung der Variablen var in float: float(var)

# Operatoren in Python

## Arithmetische Operationen

Die wichtigsten arithmetischen Operationen sind:

|  |  |
| --- | --- |
| = Zuweisung  + Addition  - Subtraktion  \* Multiplikation  / Division | // ganzzahliger Teil bei Division zweier ganzer Zahlen  % ganzzahliger Rest bei Division zweier ganzer Zahlen (Modulo)  += Erhöhung um  –= Verminderung um  ++ Imkrement (Erhöhung im Eins)  -- Dekrement (Verminderung um Eins) |

## Boolesche Ausdrücke

Boolesche Ausdrücke liefern entweder den Wert False oder den Wert True. Deshalb werden solche Ausdrucke oft in Selektionen und in Schleifen verwendet.

Die Vergleichsoperatoren benötigen ein Paar arithmetischer Operanden. In Python sind folgende Vergleichsoperatoren implementiert:

|  |  |
| --- | --- |
| == gleich  != ungleich  < kleiner als | > größer als  <= kleiner gleich  >= größer gleich |

Boolesche Ausdrücke lassen sich mit Hilfe boolescher Operatoren kombinieren. Die logischen Operatoren sind:

and und

or oder

Zusätzlich kann mit dem Operator

not( ... ) nicht

ein boolescher Ausdruck negiert werden.

# Algorithmische Grundstrukturen

| Name | Verbale Formulierung | Struktogramm | Python-Implementation |
| --- | --- | --- | --- |
| Sequenz/Verbundanweisung | | | |
| **Folge** | Anweisung 1  Anweisung 2  …  Anweisung n | Anweisung 1  Anweisung 2  …  Anweisung n | Anweisung 1  Anweisung 2  ...  Anweisung n |
| Selektion/Alternative/Auswahl | | | |
| **Einseitige Auswahl** | WENN Bedingung  DANN Anweisung | Bedingung  ja nein  Anweisung | **if** (Bedingung):  Anweisung |
| **Zweiseitige Auswahl** | WENN Bedingung  DANN Anweisung 1  SONST Anweisung 2 | Bedingung  ja nein  Anweisung 1  Anweisung 2 | **if** (Bedingung):  Anweisung 1  **else:**  Anweisung 2 |
| **Mehrseitige Auswahl** | FALLS Selektor  FALL 1: Anweisung 1  FALL 2: Anweisung 2  …  SONST: Anweisung n | FALLS Selektor  1 2 … SONST  Anw 1  Anw 2  …  Anw n | **if** (Bedingung):  Anweisung 1  **elif** (Bedingung):  Anweisung 2  …  **else**:  Anweisung n |
| Iteration/Repitition/Wiederholung/Schleife | | | |
| **Kopfgesteuerte Schleife** | SOLANGE Bedingung  TUE Anweisung | SOLANGE Bedingung  Anweisung | **while** (Bedingung):  Anweisung |
| **Fußgesteuerte Schleife** | WIEDERHOLE Anweisung  SOLANGE Bedingung  (ist nicht fest in Python implementiert) | SOLANGE Bedingung  Anweisung | Bedingung = not(Bedingung)  **while** (Bedingung):  Anweisung |
| **Zählschleife** | FÜR i = Start a BIS Ende e mit SCHRITT s:  TUE Anweisung | FÜR i = a BIS e SCHRITT s  Anweisung | **for** i in **range**(a, e+1, s)**:**  Anweisung |

# Python-Widgets und ihre Attribute/Methoden

|  |  |  |
| --- | --- | --- |
| **Widget** | **Beschreibung** | **Bereich / Beispiel** |
| Entry Eingabefeld | Attribute: Anzahl der sichtbaren Zeichen | ef = Entry(width=5) |
|  | Methoden:  Positionierung  Auslesen des Inhaltes  Löschen des Inhaltes | ef.place(x=20, y=30)  ef.get()  ef.delete(0,END) |
| Button Schaltfläche | Attribute:  Text und Länge setzen | bn = Button(text =“Abc“, width=5) |
|  | Methoden:  Positionierung s. o.  Breite ändern | bn.config(width = 15) |
|  | Ereignismethode  Linker Mausklick an bn binden  Ereignismethode definieren | Bn.bind(“<Button-1“, bnClick)  def bnClick(evt):  #anweisungen |
| Label  Ausgabefeld | Attribute:  Text der Ausgabe setzen | af = Label(text = “Abc“) |
|  | Methoden:  Positionierung (s. o.)  Text überschreiben | **af.config(text = “Neu“)** |