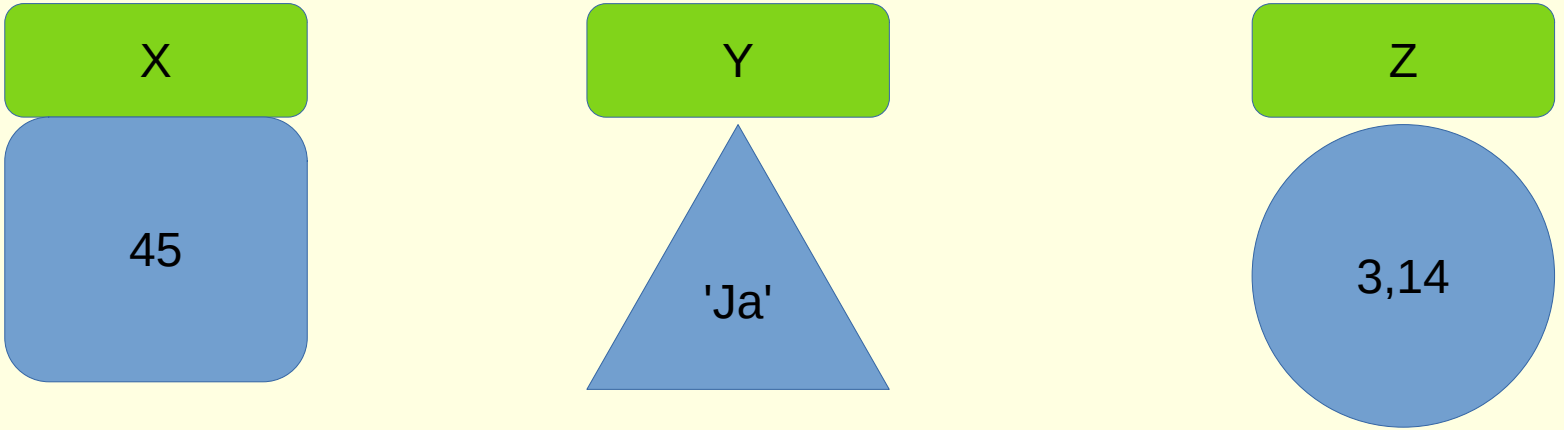
Datentypen I

Python3 – Datentypen I

Nicht jedes Ding passt in jeden Behälter



Python3 – Datentypen I

Was sind

```
5 + "vier"

2 + 2,0000

12 / 3,756
```

Python3 – Datentypen I

- Die Funktion `type()` zeigt Typ der Variable
- Typ kann sich ändern
- Hängt von gespeichertem Wert ab

```
>>> x = 5
>>> type(x)
<class 'int'>

>>> x = 2.0000
>>> type(x)
<class 'float'>

>>> x = "Baum"
>>> type(x)
<class 'str'>

```

Python3 – Datentypen I

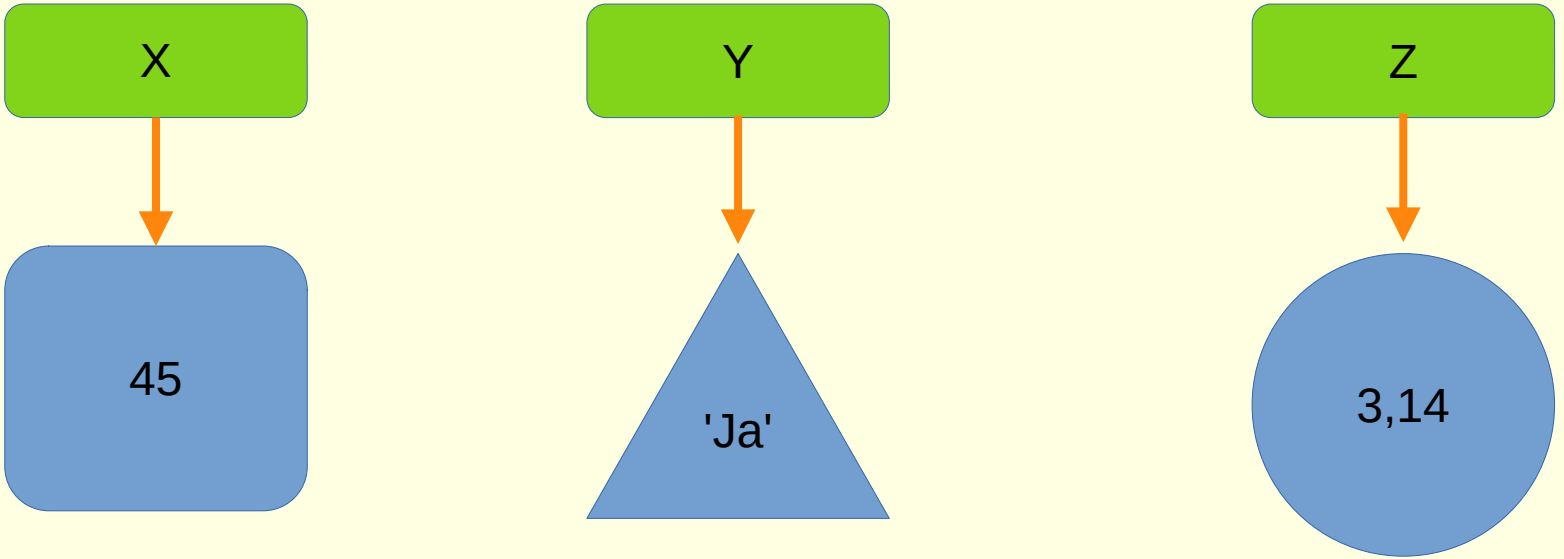
- Streng genommen: nicht Typ ändert sich
- Variable zeigt auf Behälter
- Zeigt auf neuen Behälter

  (anderen Typs)

```
>>> x = 5
>>> type(x)
<class 'int'>

>>> x = 2.0000
>>> type(x)
<class 'float'>

>>> x = "Baum"
>>> type(x)
<class 'str'>

```

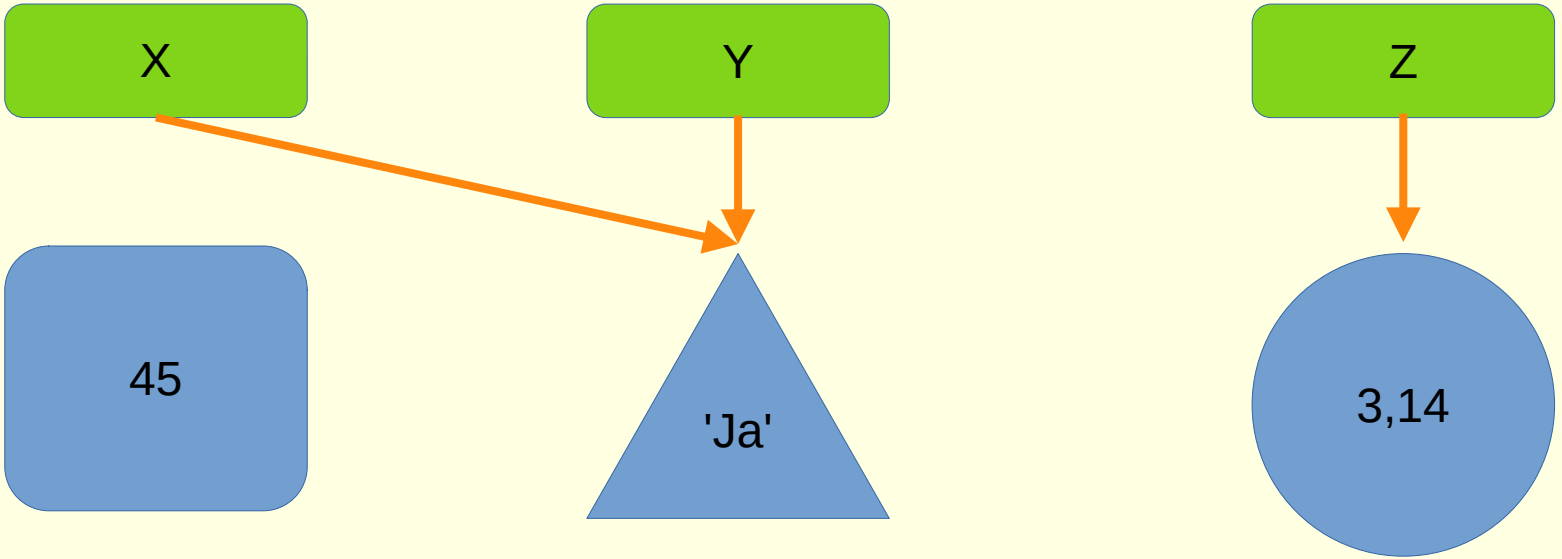
Python3 – Datentypen I

Variablen: eigentlich Zeiger



Python3 – Datentypen I

X = Y ändert nicht Inhalt sondern Ziel



Python3 – Datentypen I

- Einfachster Datentyp: Nichts / None

- z.B. wenn Wert nicht existiert:

  Höhe des kleinsten Berges über 10000m

```
1 höhe = None
2 if höhe is None:
3     print('gibt es nicht')
```

Python3 – Datentypen I

Numerische Datentypen

- int: ganze Zahlen
- float: Gleitkommazahlen
- bool: boolesche Werte
- complex: komplexe Zahlen

Python3 – Datentypen I

Datentyp int

- Ganze Zahlen
- Positiv und negativ
- Keine Beschränkung der Größe

Python3 – Datentypen I

Datentyp int -- Operationen

- Arithmetische
- Weitere s. Kapitel 11.1

```
>>> 12 + 5
17
>>> 12 - 5
7
>>> 12 * 5
60
>>> 12 / 5
2.4
>>> 12 // 5
2
>>> 12 % 5
2
```

Python3 – Datentypen I

Erweiterte Zuweisungen

Operator	Entsprechung
`x += y`	`x = x + y`
`x -= y`	`x = x - y`
`x *= y`	`x = x * y`
`x /= y`	`x = x / y`
`x %= y`	`x = x % y`
`x **= y`	`x = x ** y`
`x //= y`	`x = x // y`

Python3 – Datentypen I

Vergleichsoperatoren

Operator	Wert
`x == y`	wahr, wenn x und y gleich sind
`x != y`	wahr, wenn x und y verschieden sind
`x < y`	wahr, wenn x kleiner ist als y
`x > y`	wahr, wenn x größer ist als y
`x <= y`	wahr, wenn x kleiner oder gleich y ist
`x >= y`	wahr, wenn x größer oder gleich y ist

"wahr" heißt "boolescher Wert True", ansonsten False

Python3 – Datentypen I

Bindigkeit (Auswertungsreihenfolge)

Mit Klammern:

Reihenfolge

festlegen

```
>>> ( 3 + 4 ) * 5
35
>>> 3 + ( 4 * 5 )
23
>>> 3 + 4 * 5

   ???
```

Python3 – Datentypen I

Ohne Klammern:

- ***** bindet stärker als **+**

Python3 – Datentypen I

Ohne Klammern:

- * bindet stärker als +
- s. Buch S. 197, 198

Operator	Übliche Bedeutung
`x ** y`	`y`-te Potenz von `x`

`+x`
`-x`
`~x`
	positives Vorzeichen
	negatives Vorzeichen
	bitweises Komplement von `x`

`x * y`
`x / y`
`x % y`
`x // y`
`x @ y`
	Produkt von `x` und `y`
	Quotient von `x` und `y`
	Rest bei ganzzahliger Division von `x` durch `y`
	ganzzahlige Division von `x` durch `y`
	Matrizenmultiplikation von `x` und `y`

`x + y` `x - y`	Addition von `x` und `y` Subtraktion von `x` und `y`
`x << n` `x >> n`	bitweise Verschiebung um `n` Stellen nach links bitweise Verschiebung um `n` Stellen nach rechts
`x & y`	bitweises UND zwischen `x` und `y`
`x ^ y`	bitweises ausschließendes ODER zwischen `x` und `y`

Python3 – Datentypen I

Bindigkeit (Auswertungsreihenfolge)

Wenn Bindigkeit gleich?

```
>>> ( 10 - 4 ) - 5
1
>>> 10 - ( 4 - 5 )
11
>>> 3 - 4 - 5

   ???
```

Python3 – Datentypen I

Bindigkeit (Auswertungsreihenfolge)

Von links nach rechts

```
>>> ( 10 - 4 ) - 5
1
>>> 10 - ( 4 - 5 )
11
>>> 10 - 4 - 5
1
```

Python3 – Datentypen I

Auswertungsreihenfolge Vergleich

```
>>>   4 < 5 < 8
```

Python3 – Datentypen I

Auswertungsreihenfolge Vergleich

```
>>>   4 < 5 < 8

4 < 5 and 5 < 8
```

Beliebig lange Ketten werden zerlegt

Python3 – Datentypen I

Konvertierung numerischer Typen

- Zu jedem Typ eine Funktion

`int()` `float()` `bool()` `complex()`

Teilweise geht Information verloren

Python3 – Datentypen I

Konvertierung numerischer Typen

```
>>> bool(0)
False
>>> bool(37)
True
>>> float(7)
7.0
>>> int(7.7)
7
>>> int(True)
1
>>> complex(True)
(1 + 0j)
```

Python3 – Datentypen I

Konvertierung numerischer Typen

Nur Konvertierung
komplexer Zahlen
kann fehlschlagen

```
>>> bool(0)
False
>>> bool(37)
True
>>> float(7)
7.0
>>> int(7.7)
7
>>> int(True)
1
>>> complex(True)
(1 + 0j)
```

Python3 – Datentypen I

Wir springen zu Kontrollstrukturen, kommen später ausführlicher zu Datentypen zurück.