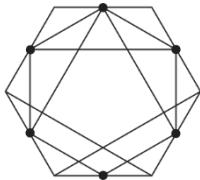
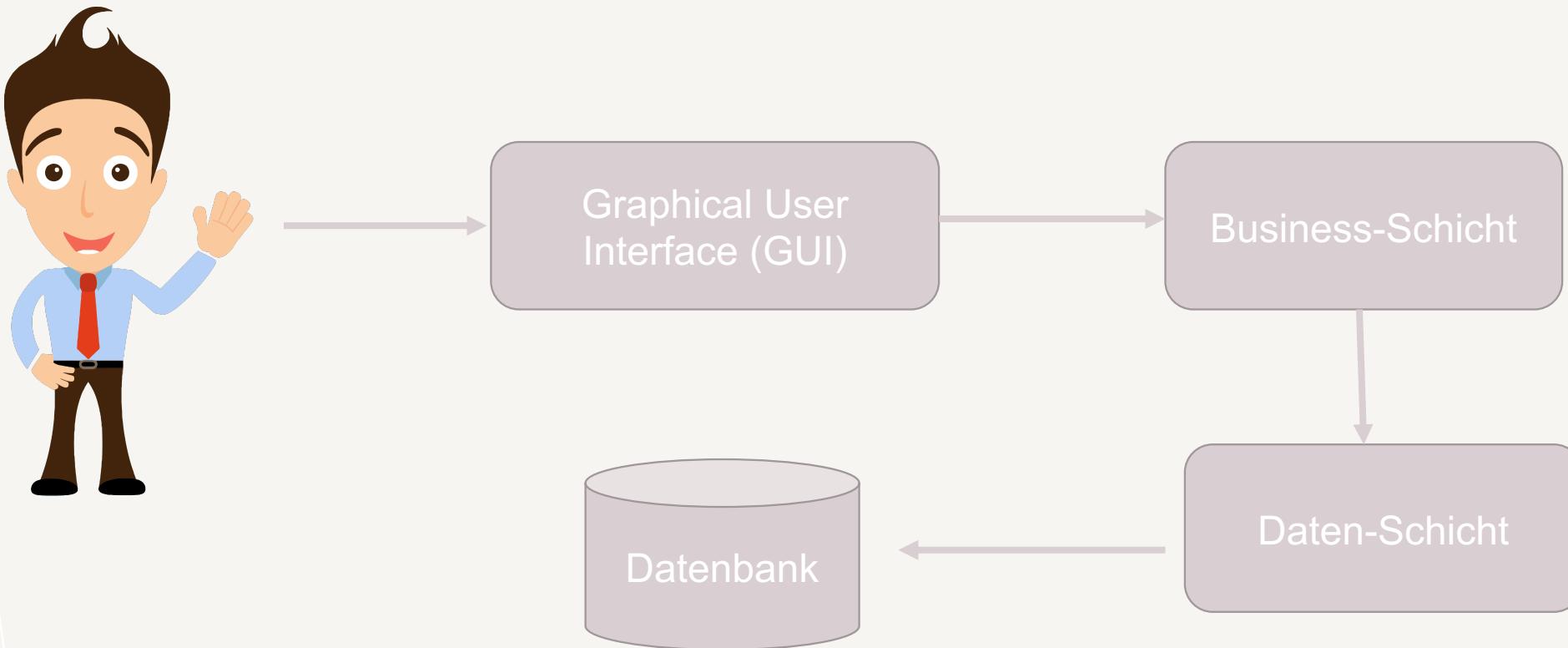


PYTHON CRASHKURS FÜR NICHT-PROGRAMMIERER



KLASSISCHE PROGRAMME



KRASHKURS ZIELE

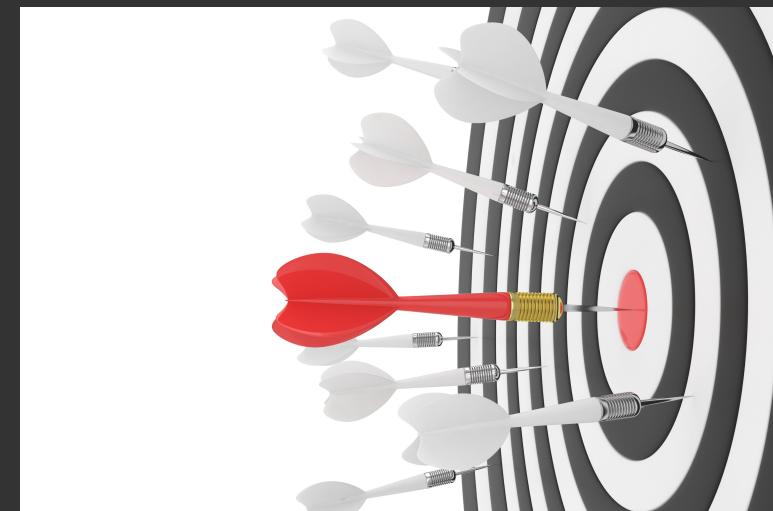
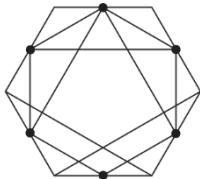
Was ist überhaupt Programmierung?

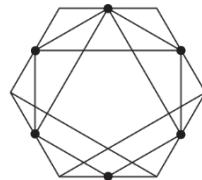
Warum Python lernen?

Datentypen in Python

Functions in Python

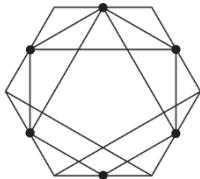
Control Flow in Python





KLASSISCHE PROGRAMME

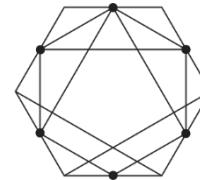
A screenshot of the Lufthansa website homepage. The page features a large banner image of a Lufthansa Boeing 747 aircraft flying in a blue sky. The Lufthansa logo is visible on the aircraft's fuselage. At the top of the page, there is a navigation bar with icons for red, yellow, and green dots, followed by browser control buttons (back, forward, search, etc.). The URL 'lufthansa.com' is displayed in the address bar. Below the banner, there is a dark blue header bar with the Lufthansa logo and a 'Menü' button. The main content area has tabs for 'Flüge' (selected), 'Flug & Hotel', 'Mietwagen', and 'Hotel'. Below these tabs, there are input fields for 'Frankfurt/Main International' (from) and 'Nach' (to), with a 'Weiter' (Continue) button. The overall design is clean and modern, with a focus on the airline's branding.



WARUM PYTHON?

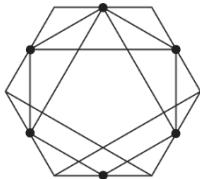
- Elegant
- Einfach
- KI Sprache!





LIVE DEMO



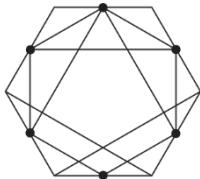


WAS IST EIN PROGRAMM?

Daten + Operationen

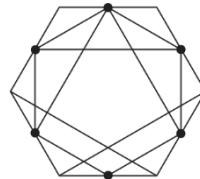
Functions + Control Flow

```
if operation == "mirror_x":  
    mirror_mod.use_x = True  
    mirror_mod.use_y = False  
    mirror_mod.use_z = False  
  
elif operation == "mirror_y":  
    mirror_mod.use_x = False  
    mirror_mod.use_y = True  
    mirror_mod.use_z = False  
  
elif operation == "mirror_z":  
    mirror_mod.use_x = False  
    mirror_mod.use_y = False  
    mirror_mod.use_z = True  
  
#selection at the end -add back the deselected mirror object  
mirror_mod.select=1  
modifier_obj.select=1  
 bpy.context.scene.objects.active = modifier_obj  
 print("Selected "+str(modifier_obj)) if modifier_obj is the active ob  
 mirror_mod.select=selected_objects[0]  
 bpy.context.scene.objects.active = selected_objects[0]
```



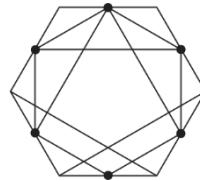
GRUNDLEGENDE DATEN

Daten	type(daten)	Beispiel
Zeichenkette	str	"Hallo" 'Hallo'
Ganze Zahlen	int	-1, 0, 1, 2, ... etc.
Gleitkommazahlen	float	-1.1, 0.3, 1.0, 1.5, ... etc.
Wahrheitswert	bool	True, False



KOMMENTARE & ZEICHENKETTEN & PRINT FUNCTION

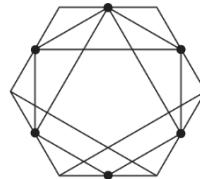
Was?	Beispiel
Kommentare	# Diese Zeile wird nicht ausgeführt
Zeichenkette über eine Zeile	"Hallo" oder 'Hallo'
Zeichenkette über mehrere Zeilen	"""Dies ist ein langer Text. Der über mehrere Zeilen geht."""
print(value1, value2,.., sep = ' ')	print("Der Wert von 1 + 1 ist", 1 + 1, ".", sep = ' ')



WORKSHOP 01: ZEICHEN KENNEN DARSTELLUNG

- Wie können Sie den String (Hello, world!) In Python darstellen?
- Wie können Sie Ihren Namen als Text (String) in Python darstellen?
- Wie können Sie (Hello, world!) Auf dem Bildschirm ausgeben?

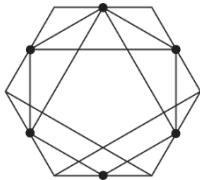




WORKSHOP: ZEICHEN KENNEN DARSTELLUNG

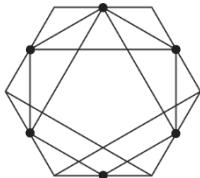
- Öffnen Sie jupyter
- Navigieren Sie zu 010 Workshop Einführung in Python
- Lösen Sie die Aufgaben unter „Einleitung“ Abschnitt





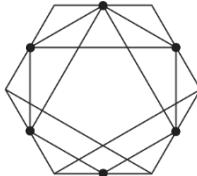
MATHEMATISCHE OPERATIONEN

Operation	Syntax	Beispiel	Ergebnis
Addition	$x + y$	$9 + 4$	13
Subtraktion	$x - y$	$9 - 4$	5
Multiplikation	$x * y$	$9 * 4$	36
Division	x / y	$9 / 4$	2.25
Divion int	$x // y$	$9 // 4$	2
Modulo	$x \% y$	$9 \% 4$	1
Potenzen	$x ** y$	$9 ** 4$	6561
Wurzel	$x ** 0.y$	$9 ** 0.5$	3.0



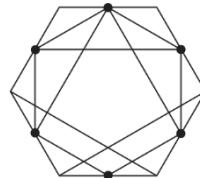
PUNKT VOR STRICH

- Erst Multiplikation oder Division, danach Addition oder Subtraktion
- $8 - 2 * 7 - 4 = ? \rightarrow = 8 - 14 - 4 \rightarrow - 10$
- $8 / 2 * 7 / 4 = ? \rightarrow = 4 * 7 / 4 \rightarrow = 32 / 7 \rightarrow 7.0$
- $8 / 2 * 7 + 4 = ? \rightarrow = 4 * 7 + 4 \rightarrow = 32 + 4 \rightarrow 32.0$



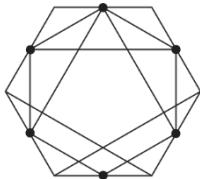
KLAMMER-RECHNUNG

- Klammer hebt Punkt vor Strich Regel auf!
- $8 / 2 * 7 + 4 = ? \rightarrow = 4 * 7 + 4 \rightarrow = 32 + 4 \rightarrow 32.0$
- $(8 / 2) * (7 + 4) = ? \rightarrow = (4) * (7+4) \rightarrow = (4) * (11) \rightarrow 44.0$



RECHTS ASSOZIATION

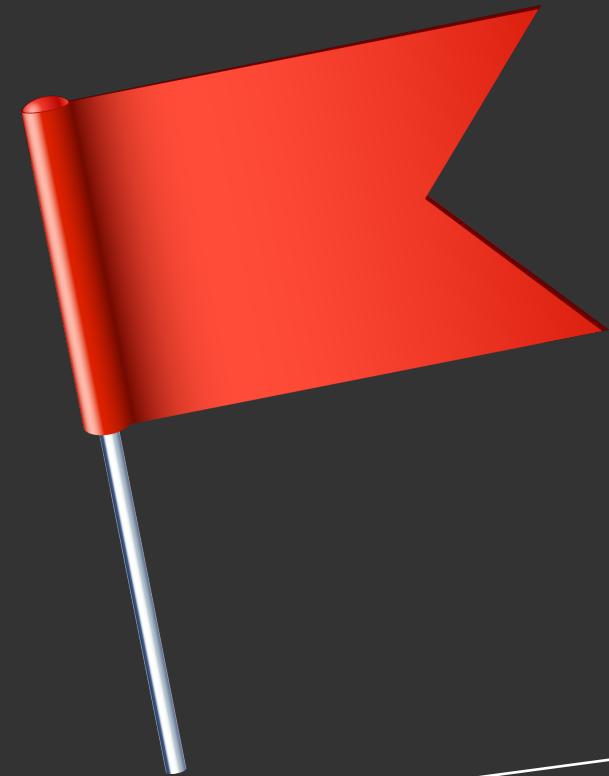
- Alle Operationen sind links Assoziation bis auf $**$ ist Rechts Assoziation
- $2 ** 3 ** 4 \rightarrow = 2 ** (3 ** 4) \rightarrow = 2 ** 81 \rightarrow 2417851639229258349412352$
- $(2 ** 3) ** 4 \rightarrow = (8) ** 4 \rightarrow = 4096$

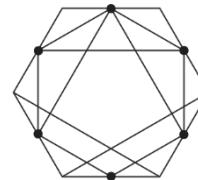


MATH OPERATIONEN NICHT FÜR BWL

$$1.1 * 100 = 110.0000\ 0000\ 0001$$

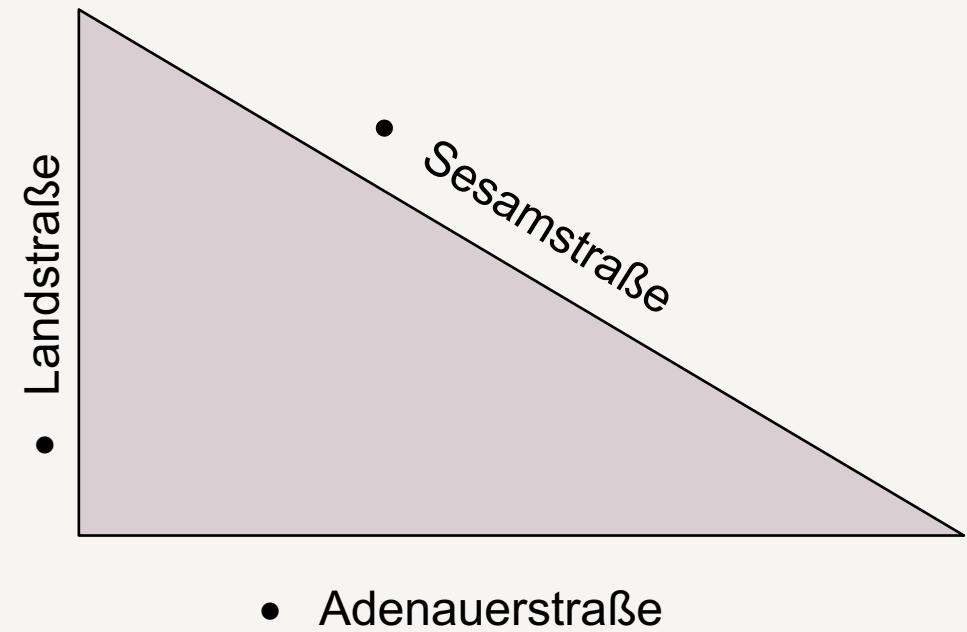
```
from decimal import Decimal  
Decimal('1.1') * 100  
output[xx]: Decimal('110.0')
```

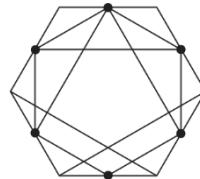




WORKSHOP : PYTHON ALS TASCHEN-RECHNER

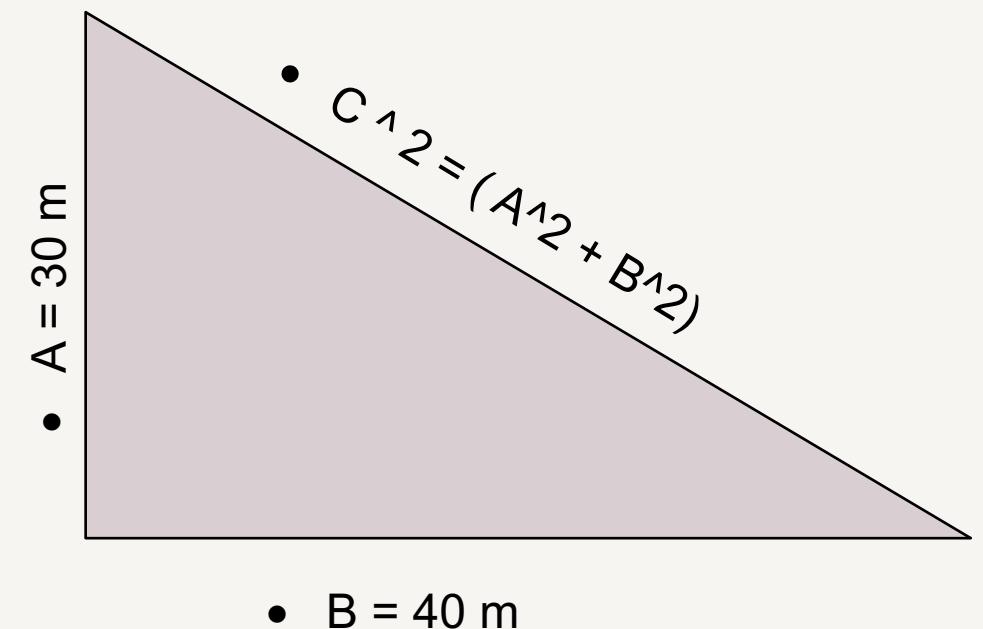
- Sie haben ein Grundstück
- Wollen einen Zaun bauen
- Landstrasse = 30 m
- Adenauerstrasse = 40 m
- Wie lang soll Ihr Zaun sein?

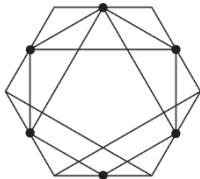




WORKSHOP 01 TIPP: DER SATZ DES PYTHAGORAS

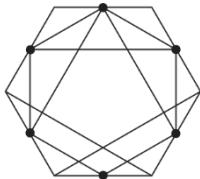
- Die Summe der quadrierten Katheten (a und b)
- Ist gleich dem Quadrat der Hypotenuse (c)





WAS SIND VARIABLEN?

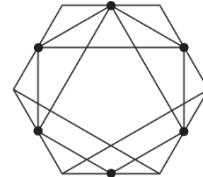




WAS IST EINE VARIABLE?

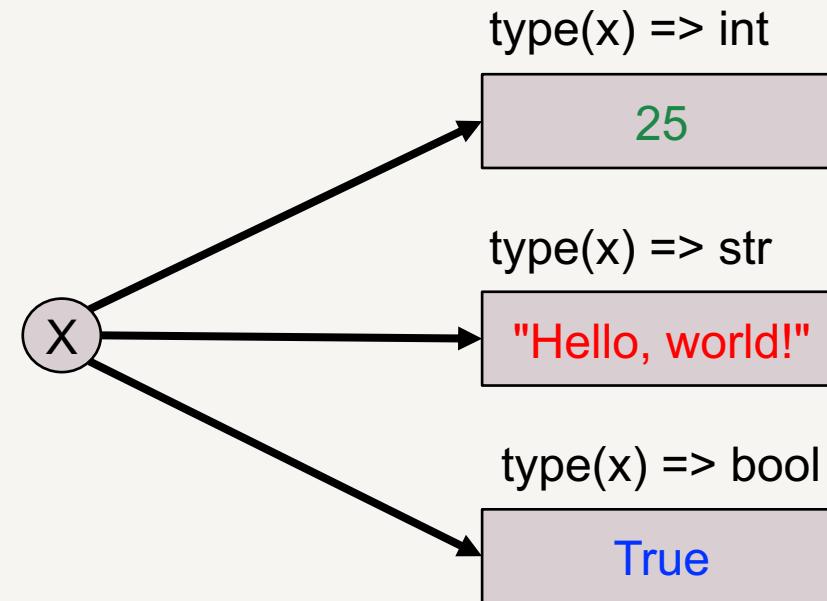
- Platzhalter
- Speichert Daten
- Hat einen Namen
- Hat einen Wert

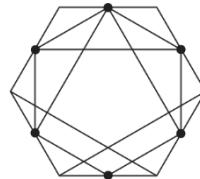




VARIABLEN IN DYNAMISCHE SPRACHEN

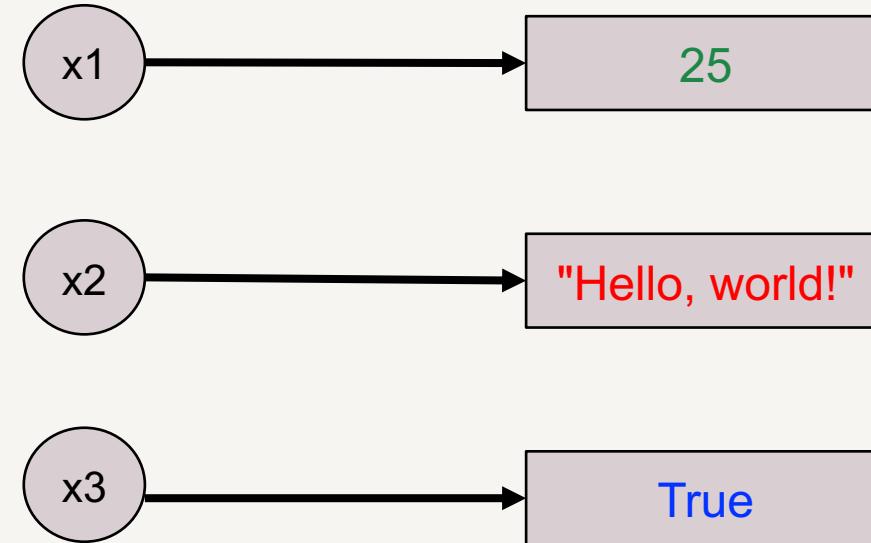
- `X = 25`
- `X = "Hello, world!"`
- `X = True`

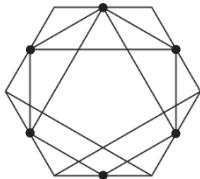




VARIABLEN IN STARK TYPESIERTE SPRACHEN

- `int x1 = 25 ;`
- `String x2 = "Hello, world!" ;`
- `boolean x3 = true ;`



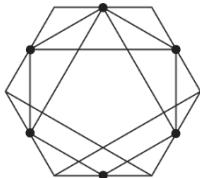


VARIABLE IN PYTHON

- Erzeugt durch `name = wert` `tasse = 100`
- Gelesen durch `name` `tasse`
- Geändert durch `name = wert` `tasse = 70`

Erzeugen und Ändern von Variablen sind *Anweisungen*





EIGENSCHAFTEN VON VARIABLEN

IN PYTHON

Kann Werte mit beliebigem Datentyp speichern (dynamisch getypt)

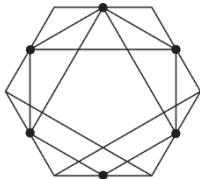
Variablen müssen erzeugt worden sein, bevor sie verwendet werden

Man kann Variablen neue Werte zuweisen

```
Zähler = 5
```

```
zähler = zähler + 1
```

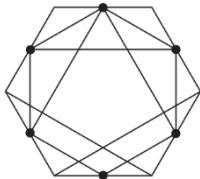




VARIABLEN NAMEN

- ❑ Fangen mit einem Buchstaben oder Unterstrich `_` an
 - ❑ Umlaute gelten auch als Buchstaben
- ❑ Können Ziffern, Buchstaben und Unterstriche `_` enthalten
- ❑ Können viele andere Unicode-Zeichen enthalten
 - ❑ Es ist aber meist besser, das zu vermeiden..
- ❑ Groß- und Kleinschreibung wird unterschieden
- ❑ A ist eine andere Variable als a

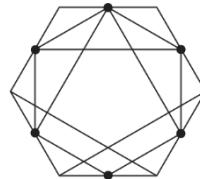




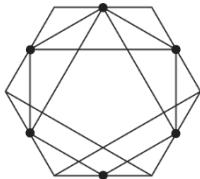
VARIABLEN STIL

- Variablennamen werden klein geschrieben
 - Außer konstanten Variablen: CONSTANT_VAR
- Bestandteile werden durch Unterstriche _ getrennt
 - Beispiel: german_song # Snake-Case
- Variablen, die mit zwei Unterstrichen anfangen und aufhören
 - Haben eine spezielle Bedeutung
 - Werden *Dunders* genannt: __class__, __name__
- Noermale Variablen sollen nicht als Dunders benannt werden





Pause ☺

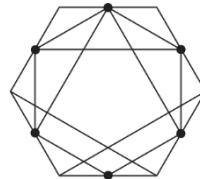


WAS IST EIN PROGRAMM?

Daten

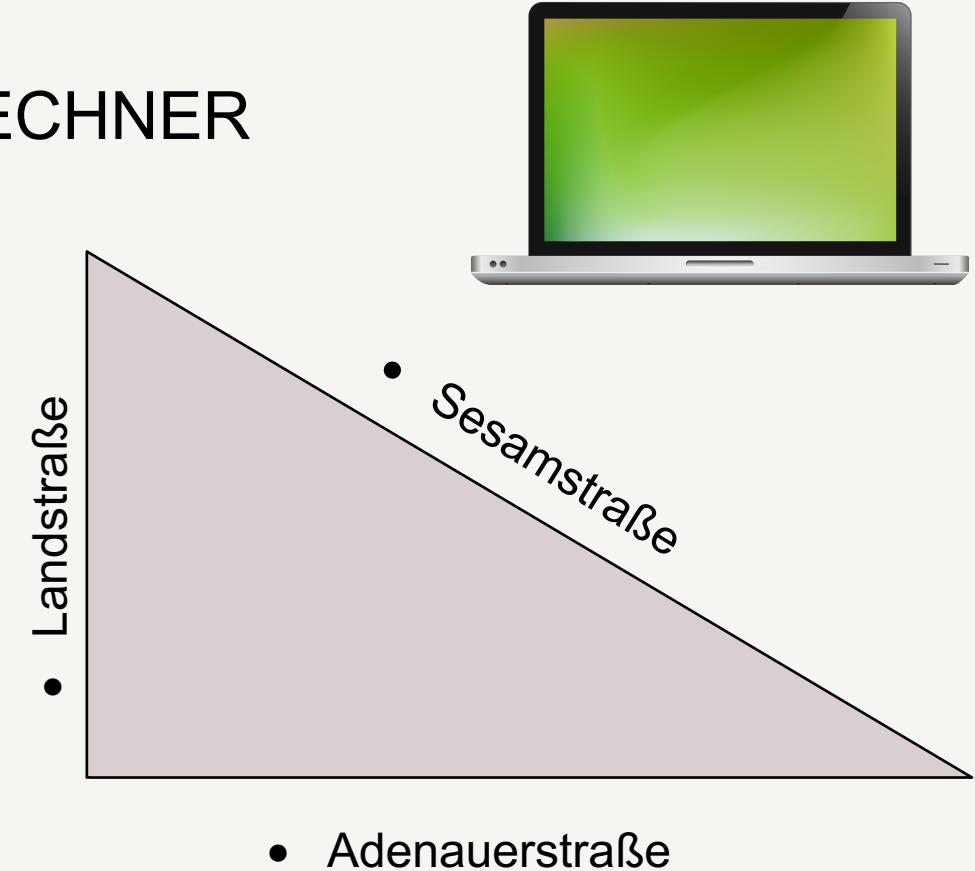
Operationen drauf!

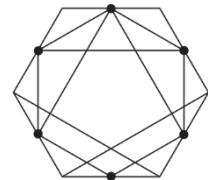
```
    #selection at the end -add back the deselected mirror modifier object
selected_obj.select = True
modifier_obj.select = False
bpy.context.scene.objects.active = modifier_obj
print("Selected" + str(modifier_obj)) # modifier ob is the active ob
    mirror_obj.select = True
    obj = bpy.context.selected_objects[0]
    bpy.ops.object.select_all(action='DESELECT')
    obj.select = True
    bpy.context.scene.objects.active = obj
    bpy.ops.object.modifier_add(type='MIRROR')
    bpy.context.object.modifiers[0].use_x = False
    bpy.context.object.modifiers[0].use_y = True
    bpy.context.object.modifiers[0].use_z = False
else:
    mirror_obj.select = True
    obj = bpy.context.selected_objects[0]
    bpy.ops.object.select_all(action='DESELECT')
    obj.select = True
    bpy.context.scene.objects.active = obj
    bpy.ops.object.modifier_add(type='MIRROR')
    bpy.context.object.modifiers[0].use_x = False
    bpy.context.object.modifiers[0].use_y = False
    bpy.context.object.modifiers[0].use_z = True
```



WORKSHOP : PYTHON ALS TASCHEN-RECHNER

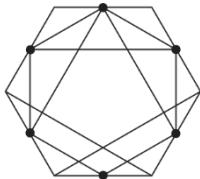
- Landstrasse = 30 m
- Adenauerstrasse = 40 m
- Wie lang soll Ihr Zaun sein?
- Lösen Sie die Aufgabe mit Variablen





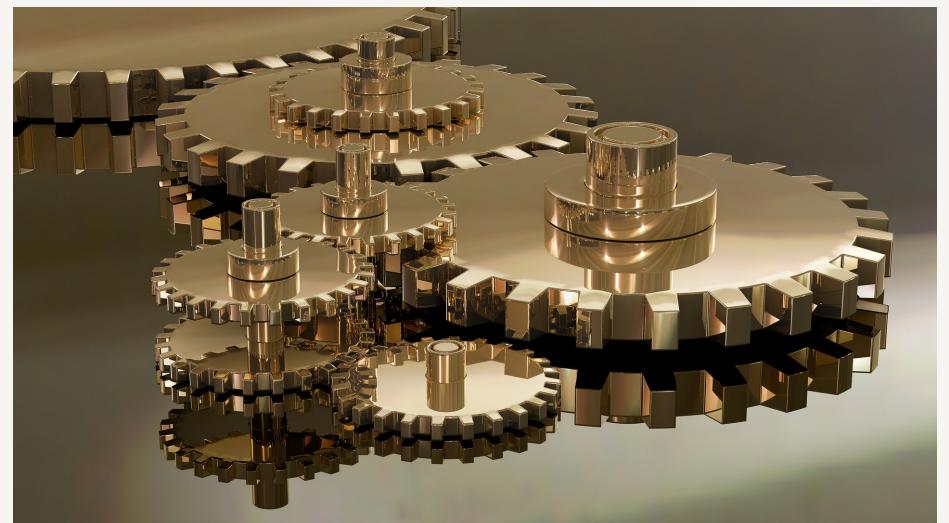
FUNCTION

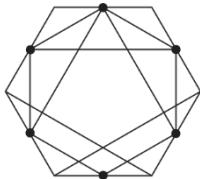




WAS IST EINE FUNKTION?

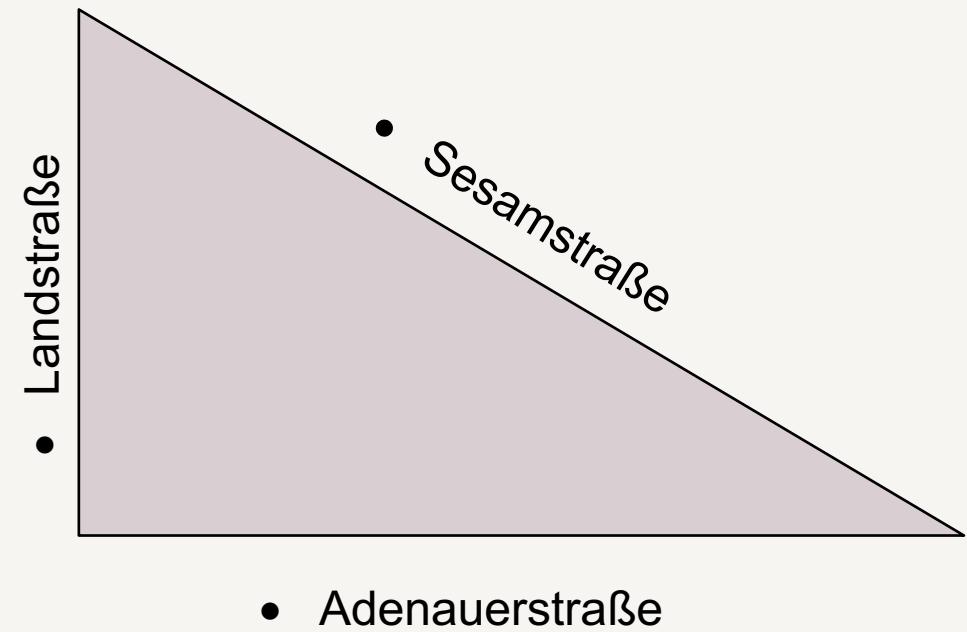
- ❑ Das Herzstück von jedem Programm
- ❑ Generalisiert Lösungen
- ❑ Implementiert Berechnungen, Algorithmen, etc.

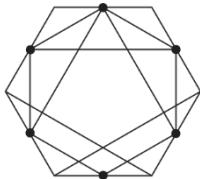




FUNKTION MOTIVATION

- Landstrasse = 30 m
- Adenauerstrasse = 40 m
- Wie lang soll Ihr Zaun sein?
- Nun weitere ähnliche Aufgabe!
- Landstraße = 70 m
- Adanuerstrasse = 115 m





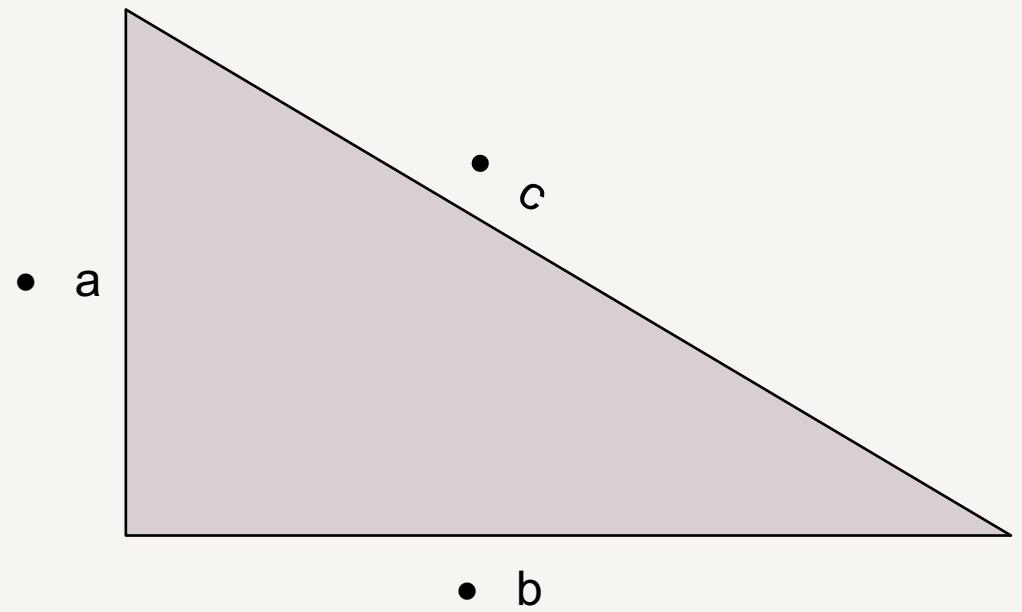
FUNKTION MOTIVATION

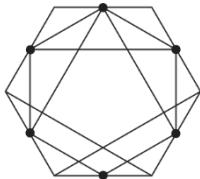
1) Funktion Definieren

```
def pythagoras(a, b):  
    quadratsumme = a ** 2 + b ** 2  
    return quadratsumme ** 0.5
```

2) Funktion Aufrufen

```
c = pythagoras(70, 135)
```



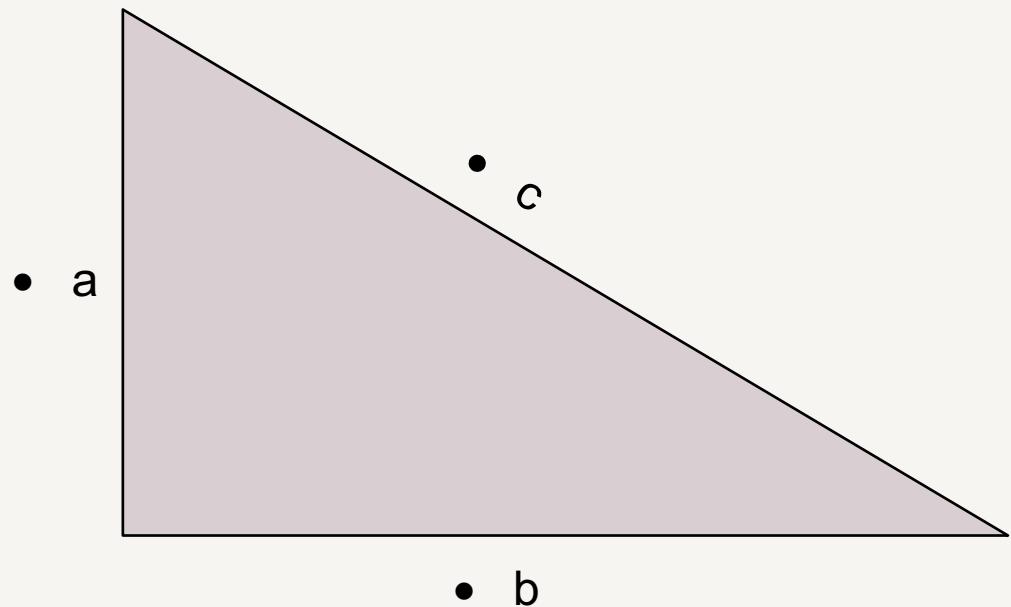


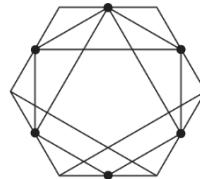
FUNKTION MOTIVATION

```
def pythagoras(a, b):  
    quadratsumme = a ** 2 + b ** 2  
    return quadratsumme ** 0.5
```

```
def gesamtlänge(x, y):  
    z = pythagoras(x, y)  
    länge = x + y + z  
    return länge
```

```
zaun = gesamtlänge(70, 135)
```

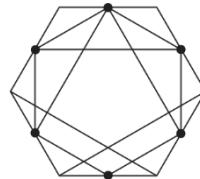




FUNCTION SYNTAX

```
def function_name(par1, parN):  
    Rumpf  
    return value
```

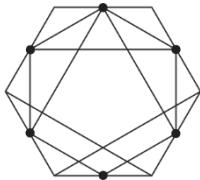
Syntax	Bedeutung	Erklärung bzw. Beispiel
def	Schlüsselwort	definiert eine Funktion
function_name	Funktionsname	calculate_salary
(par1, parN):	Parameterliste + Doppelpunkt	Kann leer sein, runde Klammer dürfen nicht fehlen!
Rumpf	Einen Tabulator eingetragen	Statements mit den Parametern, Function Calls,..
return	Schlüsselwort	Beendet die Funktion / Liefert einen Wert zurück



BEISPIEL: SAY_HELLO FUNCTION

- Hat keine Parameter
- Hat kein return-Statement

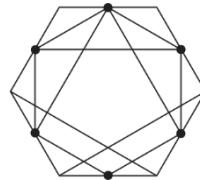
```
def say_hello():
    print("Hello, world!")
    print("Today is a great day!")
```



FUNKTIONSAUFRUF = FUNCTION CALL SYNTAX

`def function_name(par1, parN):`  `function_name(value_1, value_N)`

Syntax	Erklärung bzw. Beispiel
<code>function_name</code>	Funktionsname
<code>(value_1, value_N)</code>	Argumente des Aufrufs, in Klammern
Beispiele:	<code>pythagoras (3,4), say_hello()</code>



FUNKTIONSAUFRUF = FUNCTION CALL SYNTAX

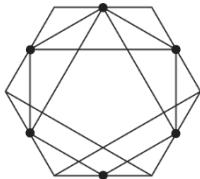
Function Call Syntax

`function_name(value_1, value_N)`

Function Call Example

`pythagoras (3 ,4), say_hello()`

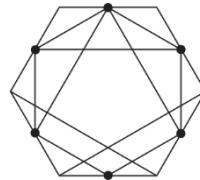
- Ein Funktionsaufruf ist ein Ausdruck:
 - Erzeugt also einen Wert
 - Gibt einen Wert zurück



WORKSHOP : PYTHON FUNCTION

- Schreiben Sie die Function `teilbar_durch`
- Überprüft ob, n durch m ohne Rest teilbar ist!
- Ist 72 durch 3 ohne Rest teilbar?



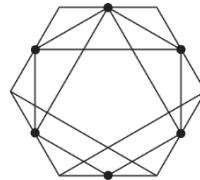


THE INPUT FUNCTION



```
variable_name = input("Bitte geben Sie den Wert ein: ")
```

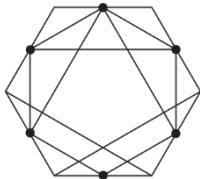
```
type(variable_name) → str
```



STR TO FLOAT



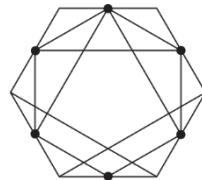
```
variable_name = input("Bitte geben Sie den Wert ein: ")  
  
float_variable = float(variable_name)
```



INPUT & STR TO FLOAT BEISPIEL

```
def konvertiere_fahrenheit_nach_celsius(fahrenheit):  
    return (fahrenheit - 32) * 5 / 9
```

```
def temperaturkonverter_1():  
    fahrenheit = input("Bitte geben Sie die Temperatur in Fahrenheit ein: ")  
    celsius = konvertiere_fahrenheit_nach_celsius(float(fahrenheit))  
    print(f"{fahrenheit}F sind {celsius}°C")
```

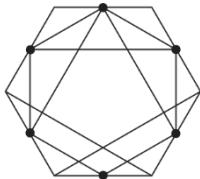


PRINT F FUNCTION

```
print(f"{{Platzhalter1} message {Platzhalter2} message")
```

```
print(f"{{fahrenheit}F sind {celsius}°C")
```

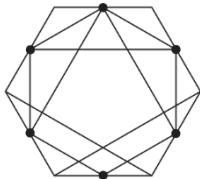




CONTROL FLOW

- ❑ Condition: if, elif, else
- ❑ Loops: while, for

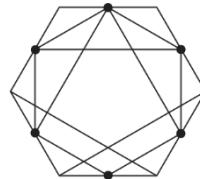




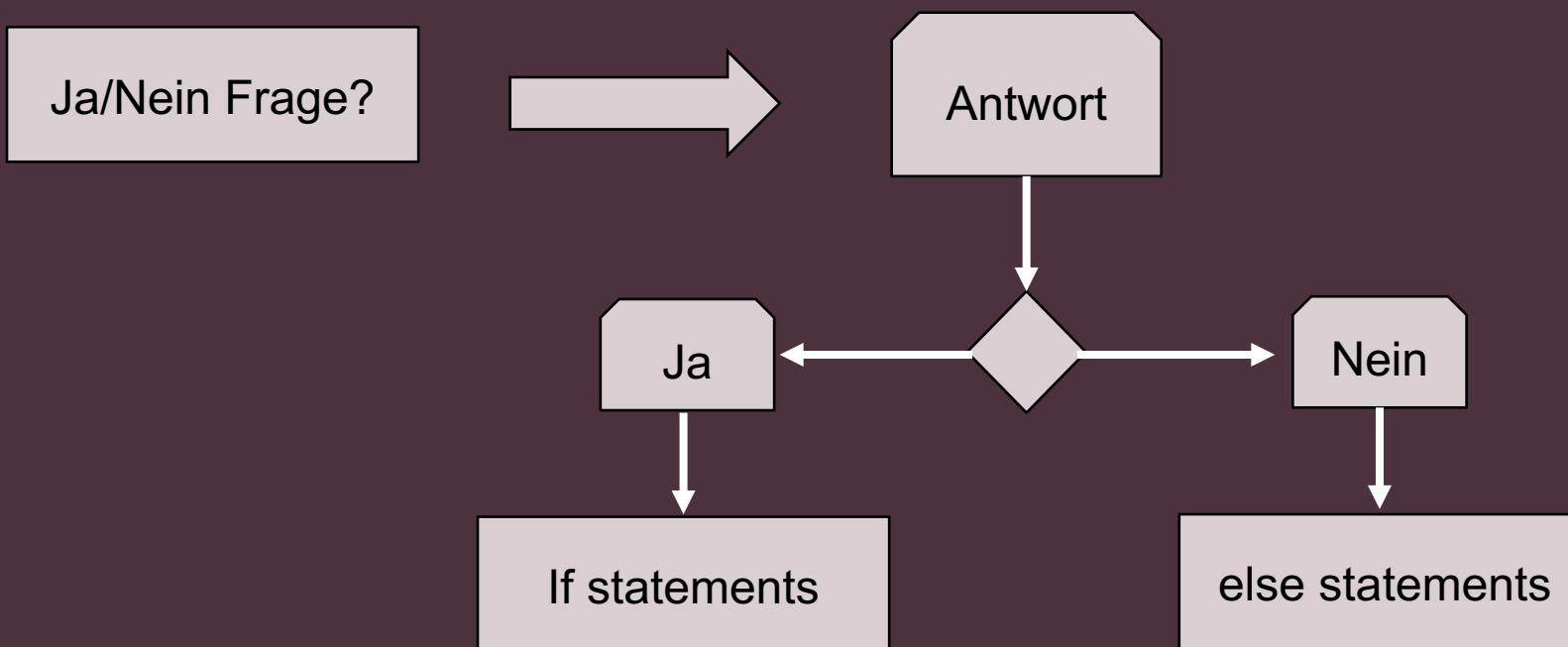
VERGLEICH OPERATIONEN

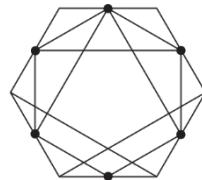
Operation	Syntax	Ergebnis	Ergebnis
<code>x == y</code>	Gleichheit von Werten	<code>1 == 1</code>	True
<code>x != y</code>	Ungleichheit von Werten	<code>5 != 7</code>	True
<code>x > y</code>	X ist größer als Y	<code>17 > 13</code>	True
<code>x >= y</code>	X ist größer oder gleich Y	<code>360 >= 390</code>	False
<code>x < y</code>	X ist kleiner als Y	<code>25 < 37</code>	True
<code>x <= y</code>	X ist kleiner oder gleich Y	<code>37 <= 37</code>	True

- Achtung: Rundungsfehler: `(2 ** 0.5) ** 2 != 2`

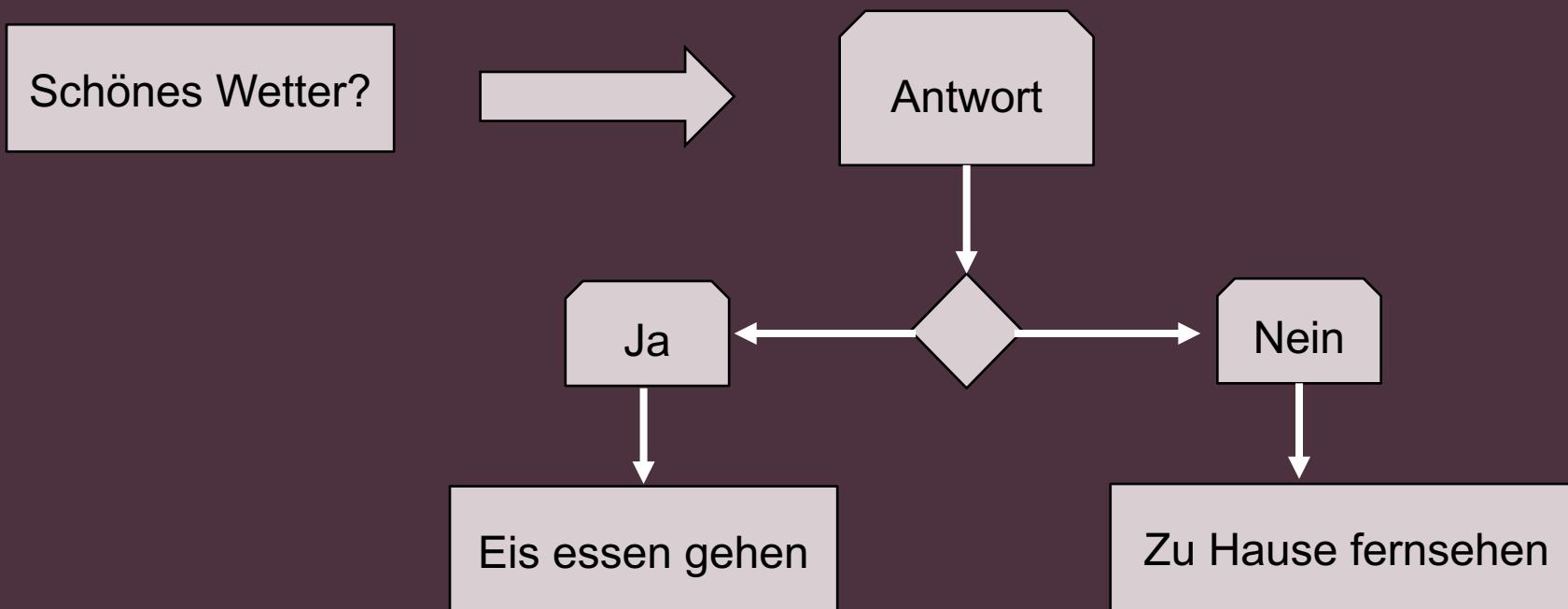


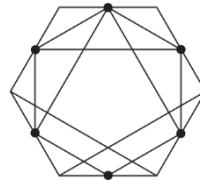
CONDITION IF, ELSE



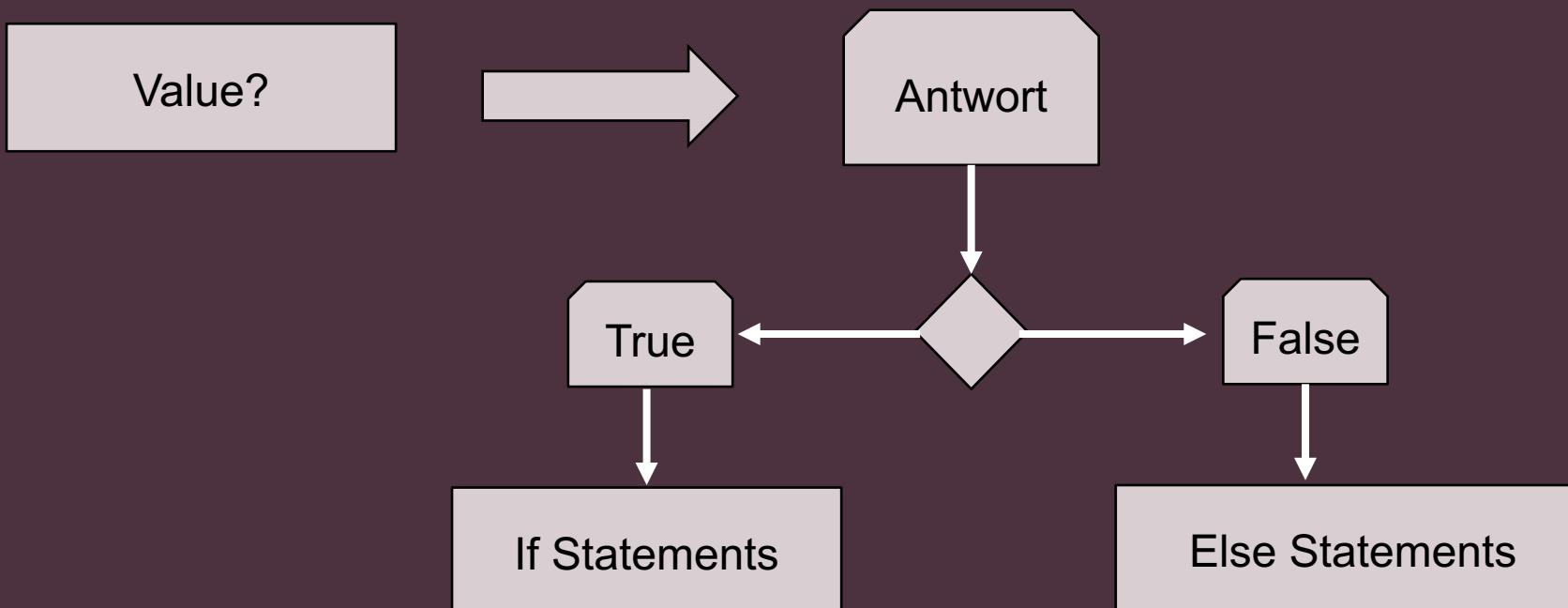


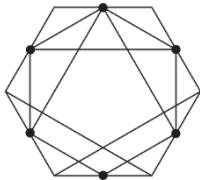
CONDITION IF, ELSE





CONDITION IF, ELSE IN DER PROGRAMMIERSPRACHE

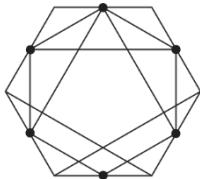




IF ELSE IN PYTHON

```
if  value:  
    if_statements  
else:  
    else_statements
```

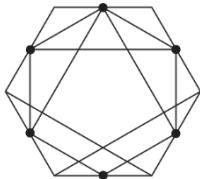
value kann	Beispiel und Regel	Ergebnis
Boolescher Ausdruck	Kann True oder False sein	$7 > 5 \Rightarrow \text{True}$. $3 < 1 \Rightarrow \text{False}$
Zahl ungleich 0	True Values: 25, -32	True
Nicht empty String	True Values: "Hola"	True
Zahl gleich 0, 0.0	False Values 0, 0.0	False
Empty String	False Value ""	False



IF / ELSE BEISPIEL 1

```
def konvertiere_fahrenheit_nach_celsius(fahrenheit):
    return (fahrenheit - 32) * 5 / 9

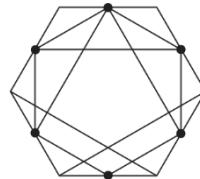
def temperaturkonverter_2():
    fahrenheit = input("Bitte geben Sie die Temperatur in Fahrenheit ein: ")
    if fahrenheit != "":
        celsius = konvertiere_fahrenheit_nach_celsius(float(fahrenheit))
        print(f"{fahrenheit}F sind {celsius}°C")
    else:
        print("Bitte geben Sie eine gültige Temperatur ein.")
```



IF / ELSE BEISPIEL 2

```
def konvertiere_fahrenheit_nach_celsius(fahrenheit):
    return (fahrenheit - 32) * 5 / 9

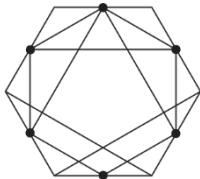
def temperaturkonverter_3():
    fahrenheit = input("Bitte geben Sie die Temperatur in Fahrenheit ein: ")
    if fahrenheit:
        celsius = konvertiere_fahrenheit_nach_celsius(float(fahrenheit))
        print(f"{fahrenheit}F sind {celsius}°C")
    else:
        print("Bitte geben Sie eine gültige Temperatur ein.")
```



WORKSHOP : IF ELSE

- Schreiben Sie die Function `was_tun`
- Fragt Sie, ob das Wetter schön ist
- Wenn Sie mit Ja antworten
- Dann gibt aus: Huraa..Lass uns Eis essen gehen
- Wenn Sie mit Nein antworten
- Dann gibt aus: **Schade Marmelade!** Lass uns zu Hause TV ansehen





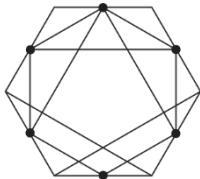
IF ELIF ELSE

Mehrere Möglichkeiten

Die sich gegenseitig ausschließen!

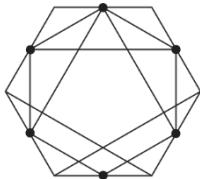
Beispiel: Note: 1.0, 1.3, 1.7, ..bis 5.0





IF ELIF ELSE

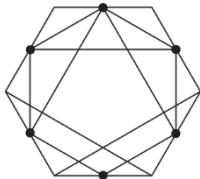
```
def bewerte_deine_note():
    note = input("Bitte geben Sie Ihre Note ein: ")
    if note < 1.7 :
        print("sehr gut")
    elif note < 2.7 :
        print("gut")
    elif note < 3.3 :
        print("befriedegend")
    elif note <= 4 :
        print("ausreichend!")
    else :
        print("Katastropheeee!")
```



WORKSHOP : IF ELIF ELSE

- Schreiben Sie die Function `berechne_mitgliedschaft`
- Fragt Sie nach Ihrem Gehalt ab
- Gehalt > 40.000 => Mitgliedschaft 500 Euro
- Gehalt zwischen 40 und 30 T => Mitgliedschaft 400 Euro
- Gehalt zwischen 30 und 20 T => Mitgliedschaft 300 Euro
- Gehalt zwischen 20 und 10 T => Mitgliedschaft 200 Euro
- Gehalt unter 10 T => Sie armer! Gehen Sie zum Sozialamt!





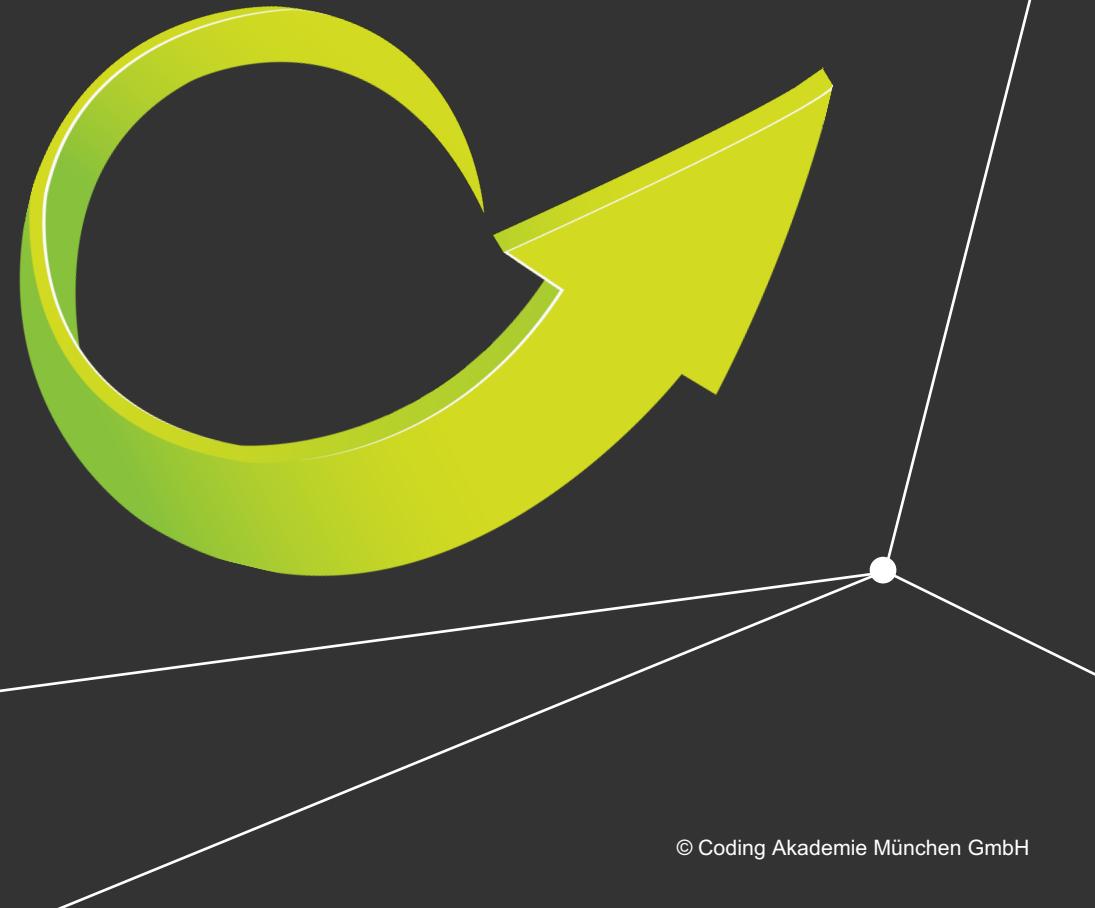
WHILE LOOP

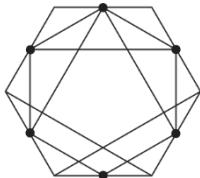
Wiederholt dieselben Anweisungen N mal

In Abhängigkeit von einem (bool) Ausdruck

Ist der Ausdruck True, wird wiederholt

Wird der Ausdruck False, hört die Schleife auf!



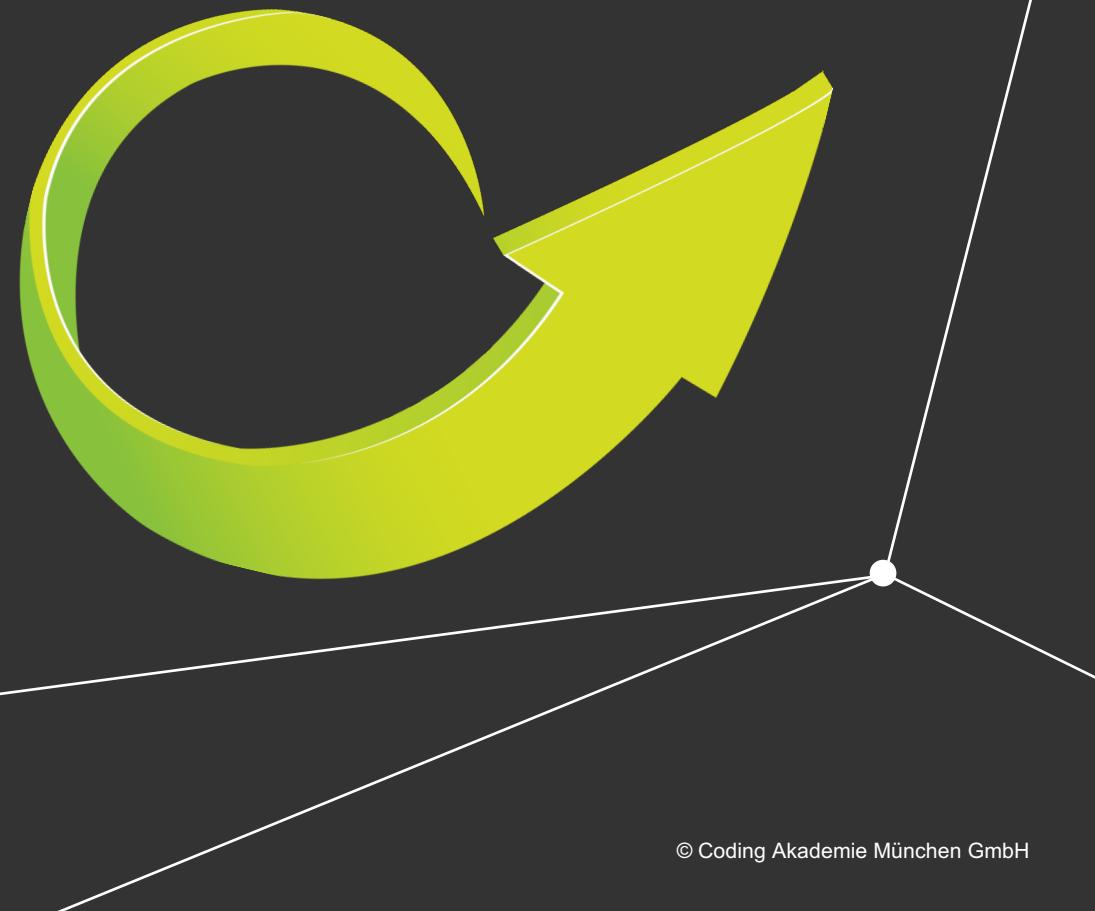


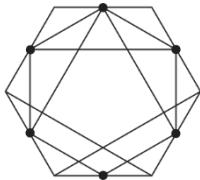
WHILE LOOP

while Abbruchsbedingung :

 statements

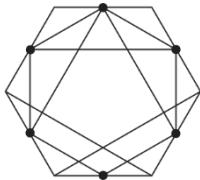
 Abbruchsbedingung update





WHILE LOOP

```
number = 0
while number < 3:
    print(f"Durchlauf {number}")
    number += 1
```



WORKSHOP : WHILE LOOPE

- Schreiben Sie eine While Schleife
- Welche die Zahlen von 0 bis 100 durchläuft
- Und nur die gerade Zahlen ausgibt!





VIELEN DANK FÜR IHRE AUFMERKSAMKEIT.