

# Algorithmen I

# Einführung in Algorithmen

Lektion 1

# Was ist ein Algorithmus?

- Definition **Algorithmus**:

- Ein Algorithmus ist eine **Abfolge eindeutiger Handlungsanweisung** für die Lösung von Problemen.
- **Eindeutig** bedeutet, dass jeder Einzelschritt zu 100% klar ist und *keinen Interpretationsspielraum* lässt.
- Algorithmen können in **verschiedenen Sprachen** formuliert werden wie menschliche Sprachen oder Programmiersprachen.
- Für die Problemlösung wird eine bestimmte **Eingabe** (Input) Schritt für Schritt in eine bestimmte **Ausgabe** (Output) überführt.

**Analogie:** Algorithmus ist wie ein (sehr klares) **Kochrezept**

# Beispiel: Karamell-Bonbon

100 g Zucker und 2 EL Wasser in Pfanne geben. Bei mittlerer Hitze erwärmen, bis der Zucker goldbraun ist. Das Caramel auf geölte Alufolie geben und auskühlen lassen.

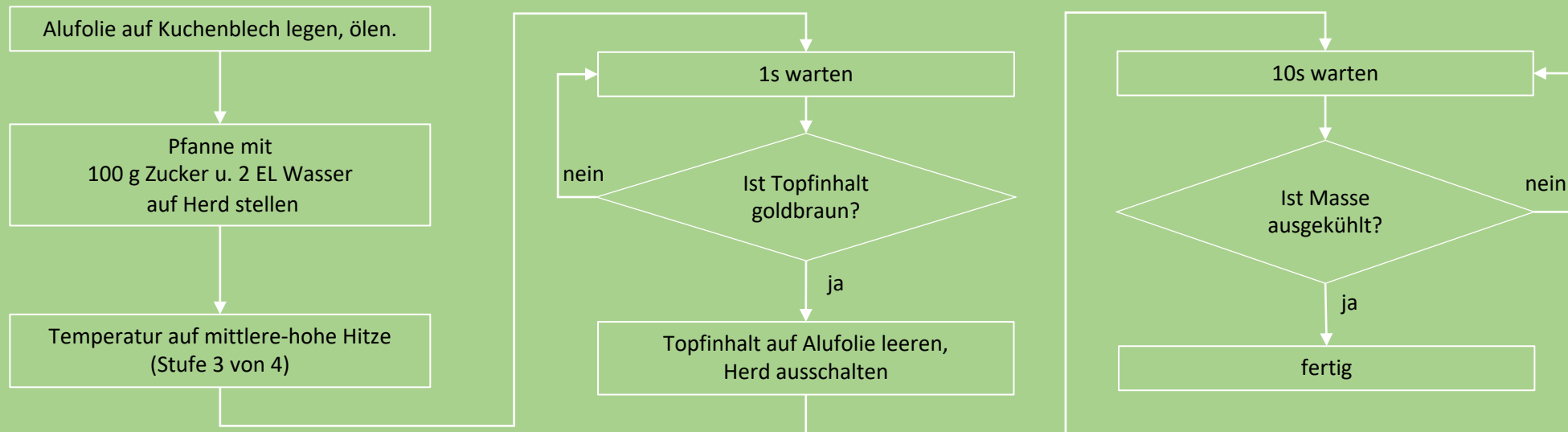
## Menschl. Sprache

```
topf.input(sugar(100,g),water(2,es))
platte.on(.75)
while not(topf.inhalt.getcolor() == "goldbraun")
    warte(1000,ms)
alufolie.input(topf.inhalt ())
platte.off()
while not(alufolie.inhalt.gettempdeg() < 20)
    warte(10000,ms)
```

## Pseudocode



## Flussdiagramm



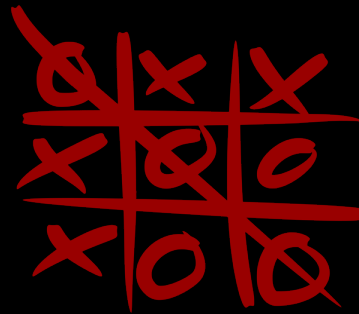
# Beispiele Algorithmen in Informatik

- **Primzahl:** Zahl wird eingegeben, Algorithmus überprüft, ob Zahl eine Primzahl ist oder nicht

2,3,5,7,11,13, ...

Wichtig z.B. für **Verschlüsselungstechniken**.

- **Spiel TicTacToe:** Computer, der gegen Menschen TicTacToe spielen kann



- Liste mit Zahlen **sortieren:**

[42 , 3, 7] → [3, 7, 42]

# Aufgabe A1: Wasserhahn

- Siehe Wiki

# Zurück zur Definition

- «Ein Algorithmus ist eine **Abfolge eindeutiger Handlungsanweisung** für die Lösung von Problemen.»
  - Der Algorithmus muss das Problem lösen ...
  - ... man muss also zum Wasserhahn kommen
  - Algorithmus darf also **keine Endlosschleifen** produzieren:
    - Sich immer im Kreis drehen
    - Immer gegen Wand laufen
    - ...
- «**Eindeutig** bedeutet, dass jeder Einzelschritt zu 100% klar ist und *keinen Interpretationsspielraum* lässt.»
  - «Drehe dich nach links» ist *nicht eindeutig*
  - «Drehe dich um 90 Grad nach links» ist *eindeutig*

# Mögliche Lösung

1. Stehe auf
2. Drehe dich um 90 Grad nach rechts
3. Laufe vorwärts, bis du eine Wand siehst
4. Drehe dich um 90 Grad nach links
5. Laufe vorwärts, bis du den Wasserhahn siehst

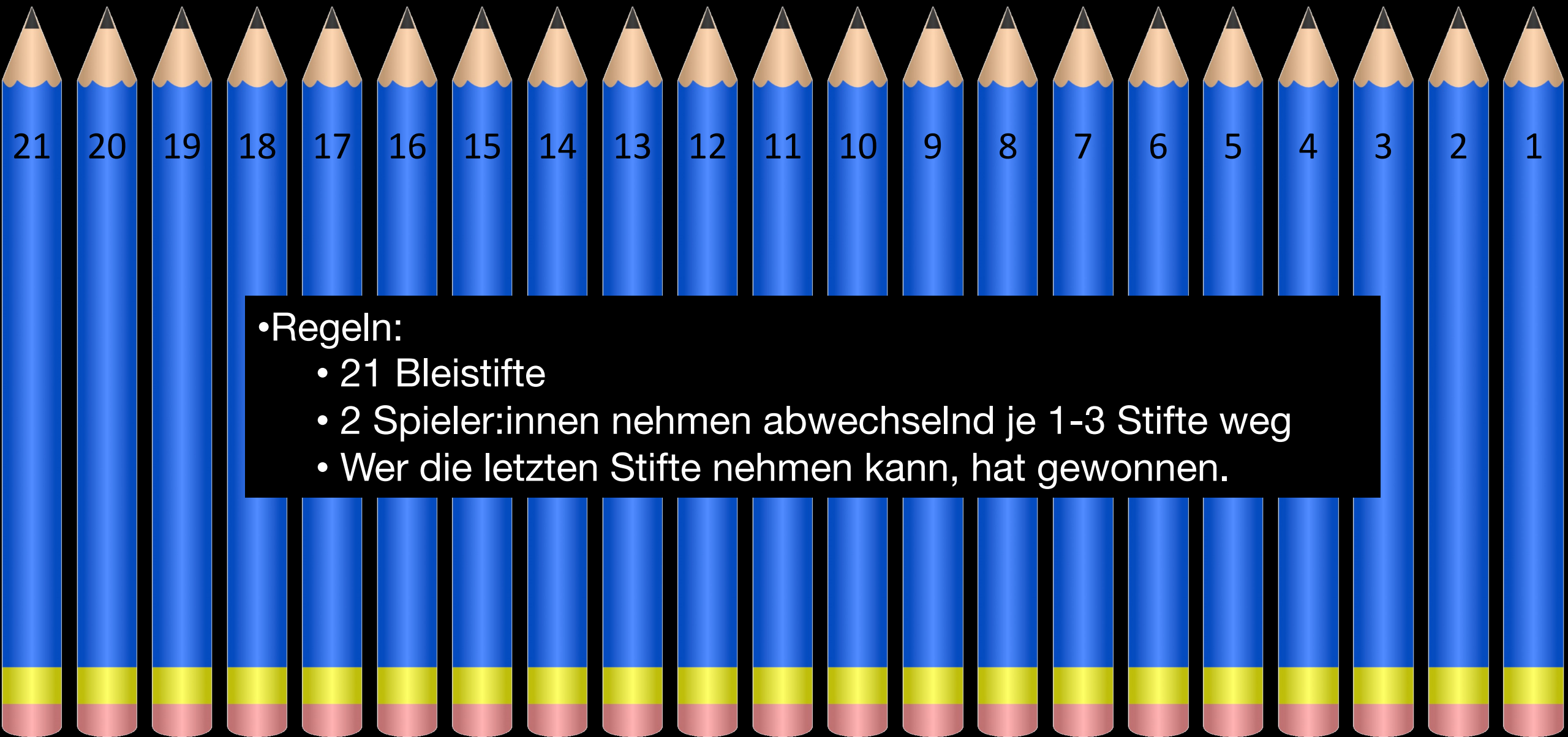
## Lektion 2

# Aufgabe A2: Subtraction Game

- Siehe Wiki

## Aufgabe A2 (Subtraction Game)

- Regeln:
  - 21 Bleistifte
  - 2 Spieler:innen nehmen abwechselnd je 1-3 Stifte weg.
  - Wer den letzten Stift nehmen kann, hat gewonnen.
- Warum ist das Spiel interessant?
  - Gibt *optimale Spielstrategie* ...
  - welche dem Computer als Algorithmus beigebracht werden kann
- Auftrag & Ziel:
  - Gegeneinander **spielen** (verwendet eure Schreiber)
  - **Optimale Spielstrategie** finden ...
  - ... und als **Algorithmus in menschlicher Sprache** formulieren



•Regeln:

- 21 Bleistifte
- 2 Spieler:innen nehmen abwechselnd je 1-3 Stifte weg
- Wer die letzten Stifte nehmen kann, hat gewonnen.

Struktogramme

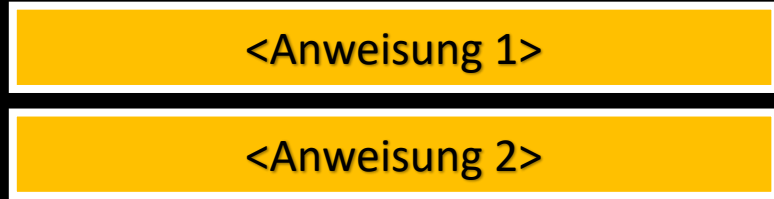
# Erinnerung: Was ist ein Algorithmus?

- Definition **Algorithmus**:
  - Ein Algorithmus ist eine **Abfolge eindeutiger Handlungsanweisung** für die Lösung von Problemen.
  - **Eindeutig** bedeutet, dass jeder Einzelschritt zu 100% klar ist und *keinen Interpretationsspielraum* lässt.
  - Algorithmen können in **verschiedenen Sprachen** formuliert werden wie menschliche Sprachen oder Programmiersprachen.
  - Für die Problemlösung wird eine bestimmte **Eingabe** (Input) Schritt für Schritt in eine bestimmte **Ausgabe** (Output) überführt.
- **Struktogramme**: Möglichkeit, um Algorithmen darzustellen, etwas zwischen normaler Sprache und Programmiersprache
- Kann damit gewissermassen *auf Papier Programmieren*

# Aufbau Struktogramm

- **3 Arten** von Elementen:

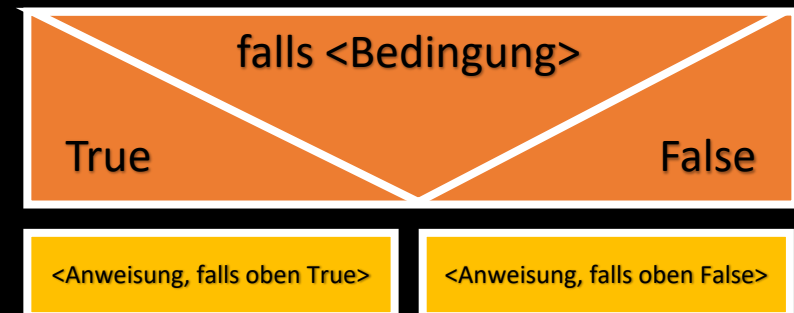
1. **Einfache Anweisungen** [Rechteck]



2. **Schleifen**



3. **Verzweigungen**



# Beispiel Wasserhahn

## 1. Einfache Anweisungen:

Stehe auf

Gehe 0.5m nach vorne

Drehe dich um 90 Grad nach links

## 2. Schleifen:

Solange keine Wand vor dir

Gehe 0.5m nach vorne

## 3. Verzweigungen:

Falls Wasserhahn vor dir

True

False

Du bist am Ziel

Gehe 0.5m nach vorne

# Aufgaben B

## Aufgabe B1 (Wasserhahn revisited)

- Schreibe deinen Algorithmus, der eine Person zum Wasserhahn bringt als Struktogramm um. Verwende nur die vorgegebenen drei Arten von Elementen.
- Erweitere Algorithmus: Die Person soll, nachdem sie den Wasserhahn erreicht hat, Wasser davon trinken.
- Es kann sein, dass Wasserhahn bereits läuft.

Tipps ▶

✎ Bearbeiten

## Aufgabe B2 (Karamell-Bonbons revisited)

Wandle das Flussdiagramm mit dem Algorithmus zur Herstellung von **Karamell-Bonbons** in ein Struktogramm um.

✎ Bearbeiten

## Aufgabe B3 (Subtraction Game revisited)

Der 'Computer' spiele gegen eine Spieler:in. Schreibe nun den Algorithmus, mit dem der Computer das Subtraction Game spielen soll, als *Struktogramm* für drei verschiedene Versionen:

1. Der Computer soll immer beginnen und natürlich immer das Spiel gewinnen.
2. Diesmal soll die Spieler:in beginnen. Der Computer soll gewinnen, sobald die Spieler:in einen Fehler macht.
3. Erweitere dein Struktogramm aus 2., so dass beide Fälle (Computer beginnt (nicht)) abgedeckt sind.

✎ Bearbeiten

Variablen, Ein- und Ausgaben und  
Co.

# Eingaben, Ausgaben und Variablen

- Wollen nun etwas mathematische Algorithmen anschauen
- Benötigen folgende Elemente:
- **Variablen:**
  - = ist Zuweisungsoperator
- **Eingaben:**
  - Runde Klammern: Art des Inputs (Zahl, Wort, Text)
- **Ausgaben:**
- Wiki: mehr Infos dazu

i = 42

x = [Eingabe] (Zahl)

Ausgabe x

# Einzelgleich (=) vs. Doppelgleich (==)

- **Einzelgleich (=):**

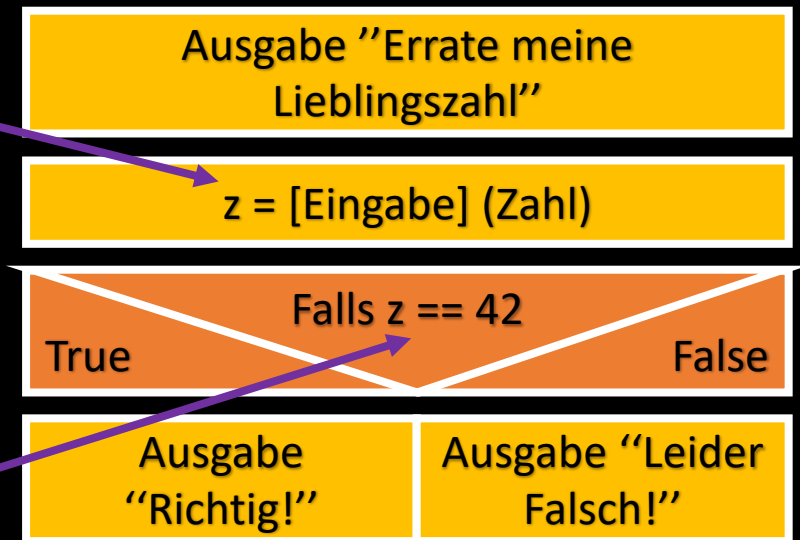
- Z.B. 'i = 3'
- Ist **Zuweisungsoperator**
- Weist Variable links den Wert rechts zu
- Macht keinen Sinn:
  - '3 = i'
  - '2i = 3'

- **Doppelgleich (==):**

- Z.B. 'i == 3'
- In Bedingungen in Falls-Verzweigungen und Solange-Schleifen
- Ist **Vergleichsoperator**

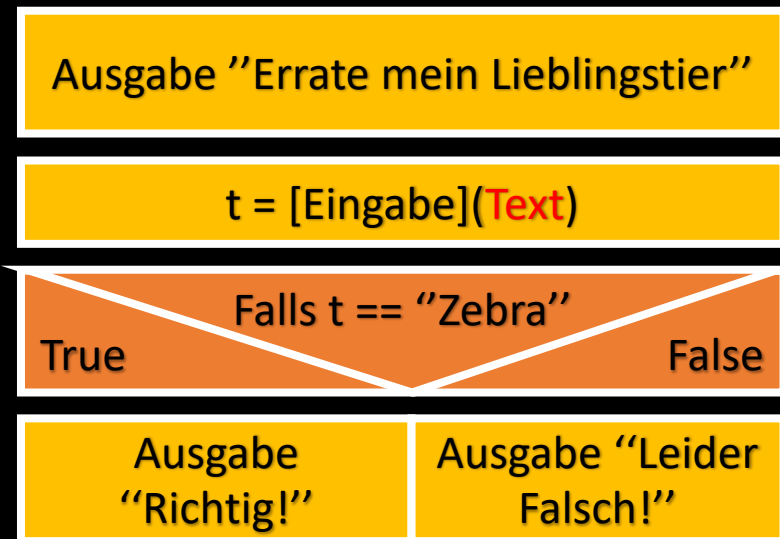
## Beispiel

Spiel, in dem Lieblingszahl von jemandem erratet werden soll



# Einzelgleich (=) vs. Doppelgleich (==)

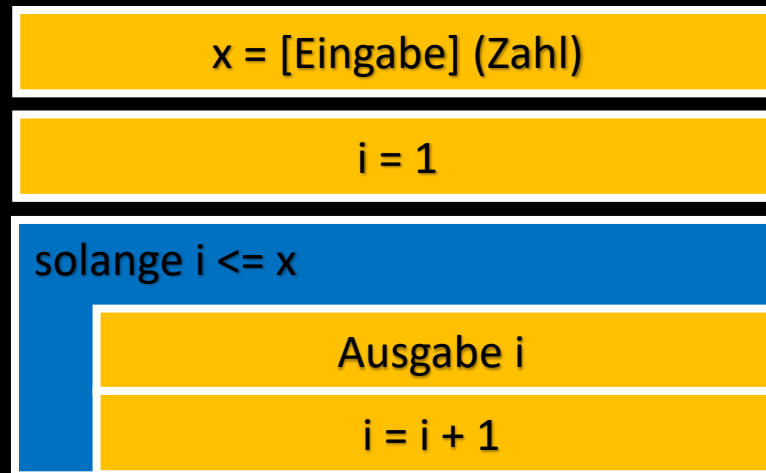
- **Auftrag:** Schreibe ähnliches Spiel, in dem jemand dein Lieblingstier erraten soll. Verwende Einzel- und Doppelgleich.



- Beachte: **Anführungs- und Schlusszeichen** bei Text um zwischen Text und Variablen unterscheiden zu können.
- Text wird **String** genannt.

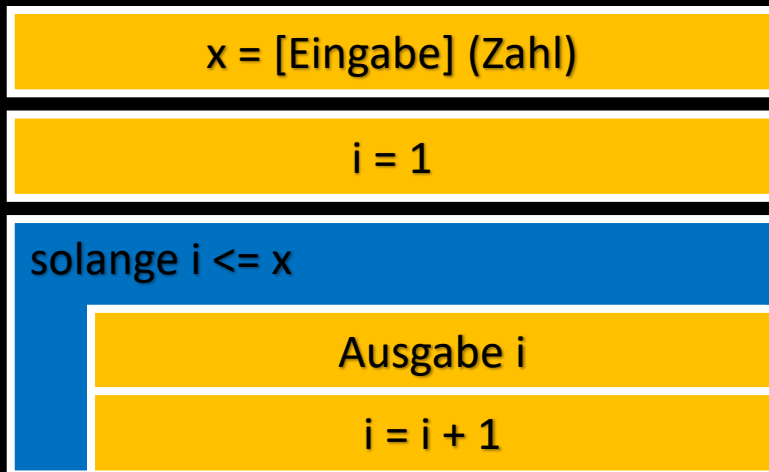
# Beispiel: Zählen

- Algorithmus:
  - Zahl eingeben (z.B. 3)
  - Algorithmus zählt von 1 auf diese Zahl hoch, gibt alle Zahlen au



# Evaluieren von Algorithmus

- Struktogramm für konkretes Beispiel durchgehen ...
- Alle Werte in Tabelle notieren



- Muss genau so notiert werden
- Genaue Regeln auf nächster Slide

| Bedingungen   | x | i | Ausgabe |
|---------------|---|---|---------|
|               | 3 |   |         |
|               |   | 1 |         |
| i <= x? True  |   |   |         |
|               |   |   | 1       |
|               |   | 2 |         |
| i <= x? True  |   |   |         |
|               |   |   | 2       |
|               |   | 3 |         |
| i <= x? True  |   |   |         |
|               |   |   | 3       |
|               |   | 4 |         |
| i <= x? False |   |   |         |

# Evaluieren von Algorithmus

- **Regeln:**

- Spalte ganz links: heisst **Bedingungen**, notiere hier Bedingungen von Schleifen und Verzweigungen und Angabe, ob erfüllt oder nicht (True / False)
- Spalte ganz rechts: heisst **Ausgabe**, notiere hier Werte, die ausgegeben werden
- Spalten dazwischen: Werte von **Variablen** in *Reihenfolge*, in der sie im Code vorkommen
- Pro ausgeführte Zeile Code, genaue **eine Zeile** in der Tabelle!

| Bedingungen   | x | i | Ausgabe |
|---------------|---|---|---------|
|               | 3 |   |         |
|               |   | 1 |         |
| i <= x? True  |   |   |         |
|               |   |   | 1       |
|               |   | 2 |         |
| i <= x? True  |   |   |         |
|               |   |   | 2       |
|               |   | 3 |         |
| i <= x? True  |   |   |         |
|               |   |   | 3       |
|               |   | 4 |         |
| i <= x? False |   |   |         |

# Aufgaben D

- Siehe Wiki