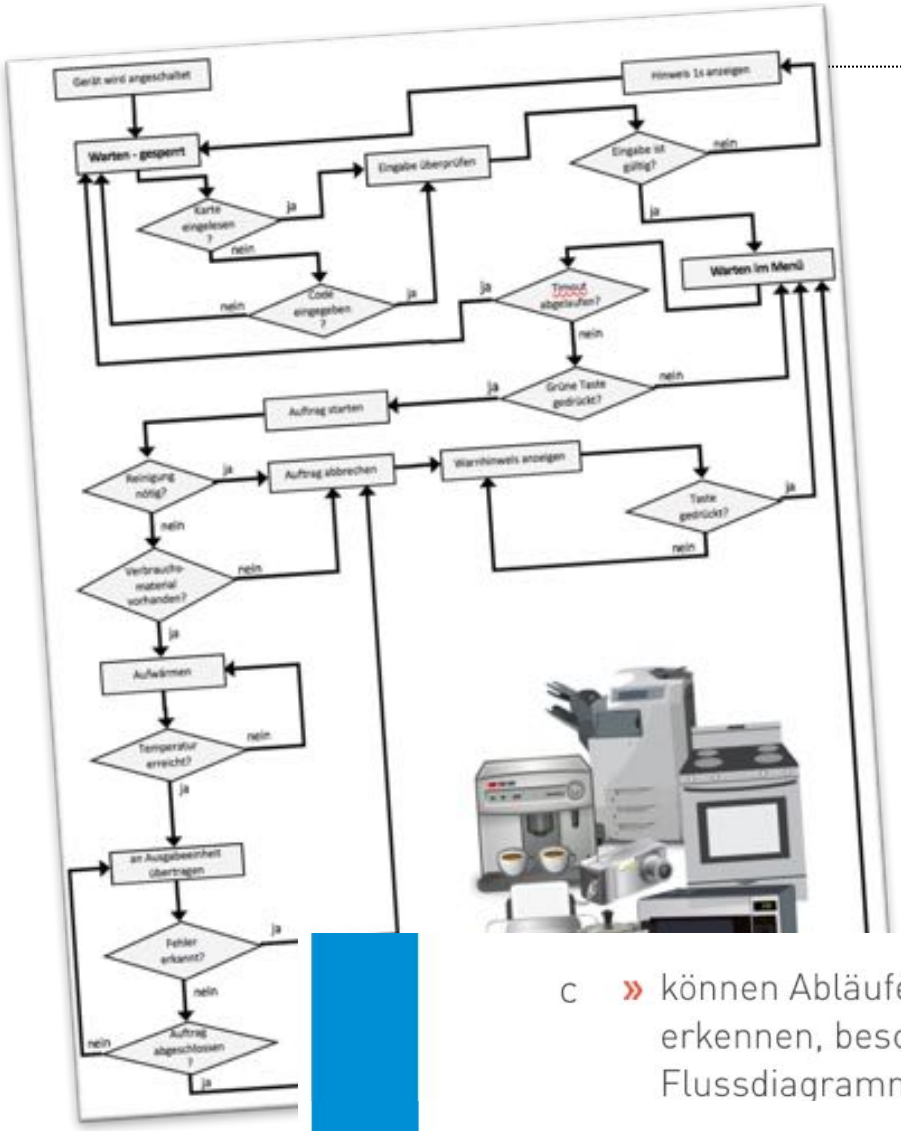
Algorithmen vergleichen: Suchen und Sortieren

Lehrplan 21

» können verschiedene Algorithmen zur Lösung desselben Problems vergleichen und beurteilen (z.B. lineare und binäre Suche, Sortierverfahren).

Flussdiagramme zum Visualisieren von Algorithmen



Arbeitsauftrag:

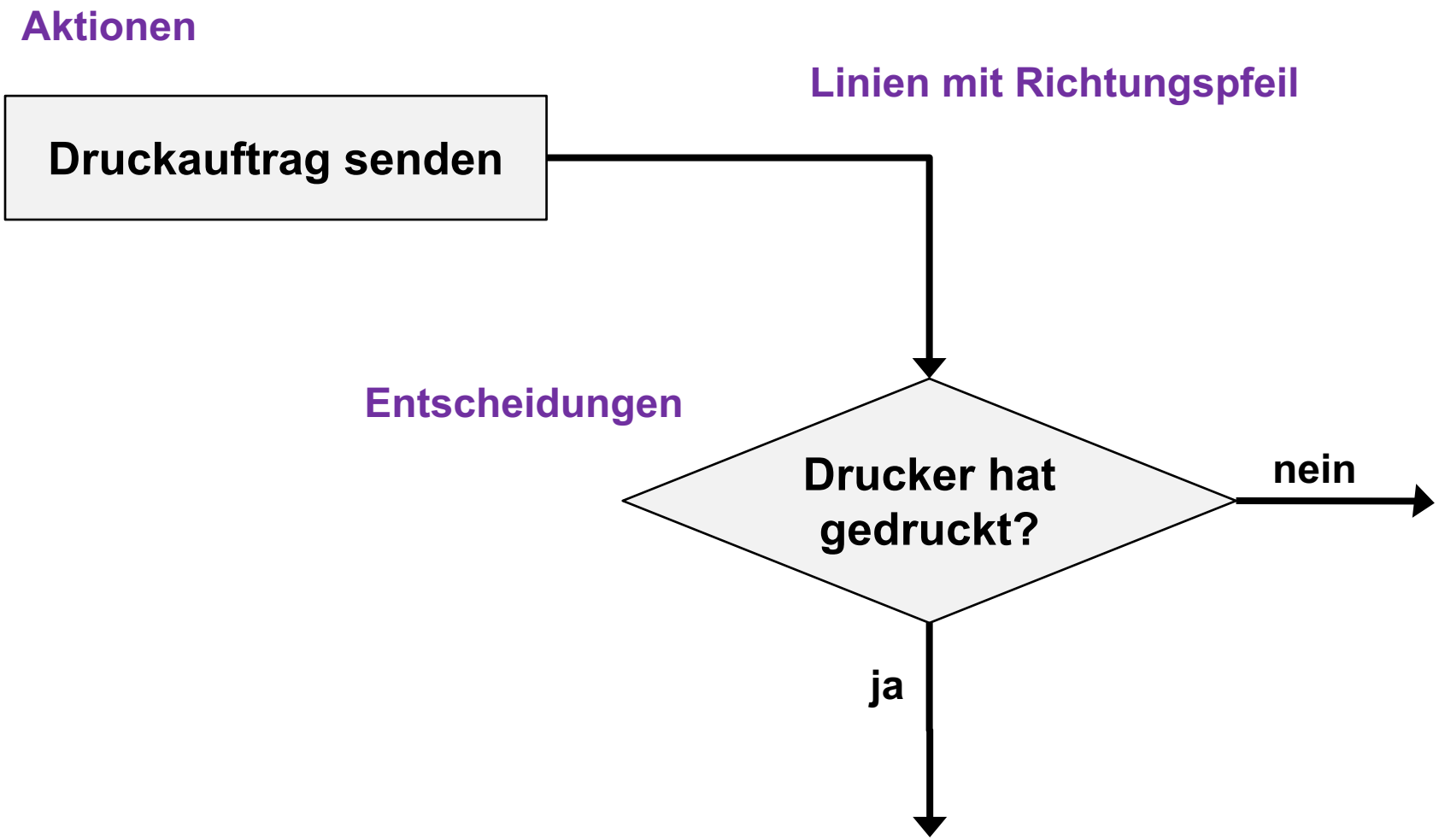
Der Ablauf welcher Maschine wird im Diagramm beschrieben?

Lehrplan 21

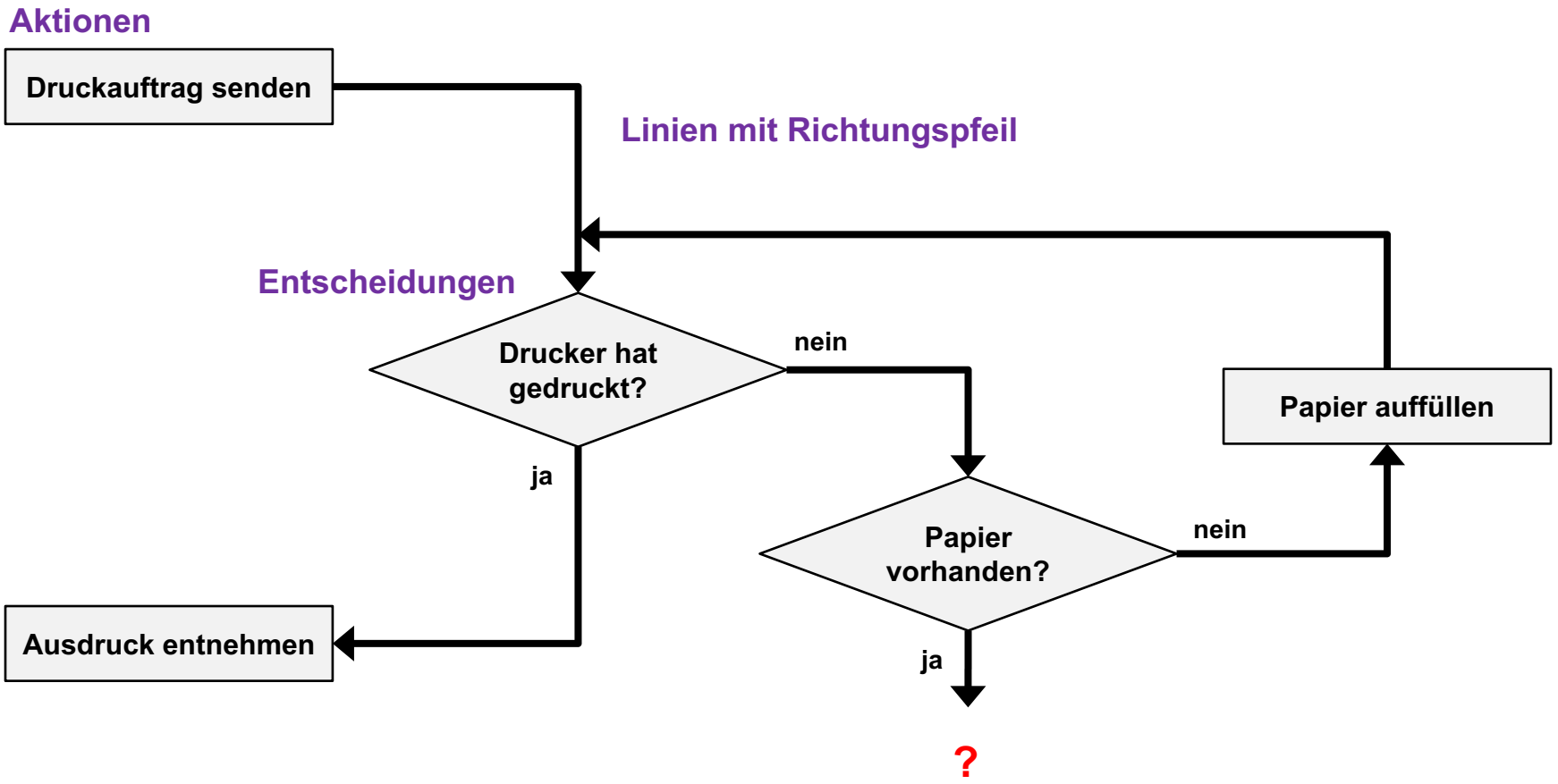
c » können Abläufe mit Schleifen und Verzweigungen aus ihrer Umwelt erkennen, beschreiben und strukturiert darstellen (z.B. mittels Flussdiagrammen).

d » können einfache Abläufe mit Schleifen, bedingten Anweisungen und Parametern lesen und manuell ausführen.

Flussdiagramm - Bausteine



Flussdiagramme zum Visualisieren von Algorithmen



Arbeitsauftrag:
Entwerfen Sie ein eigenes Flussdiagramm, wie Sie als Mensch das Problem „Drucker druckt nicht“ lösen könnten. Beziehen Sie alle möglichen Ursachen und Lösungen ein.

Beispiel: Lineare Suche vs. Binärsuche unplugged

Ein bestimmtes Element in einer grossen Datenmenge finden

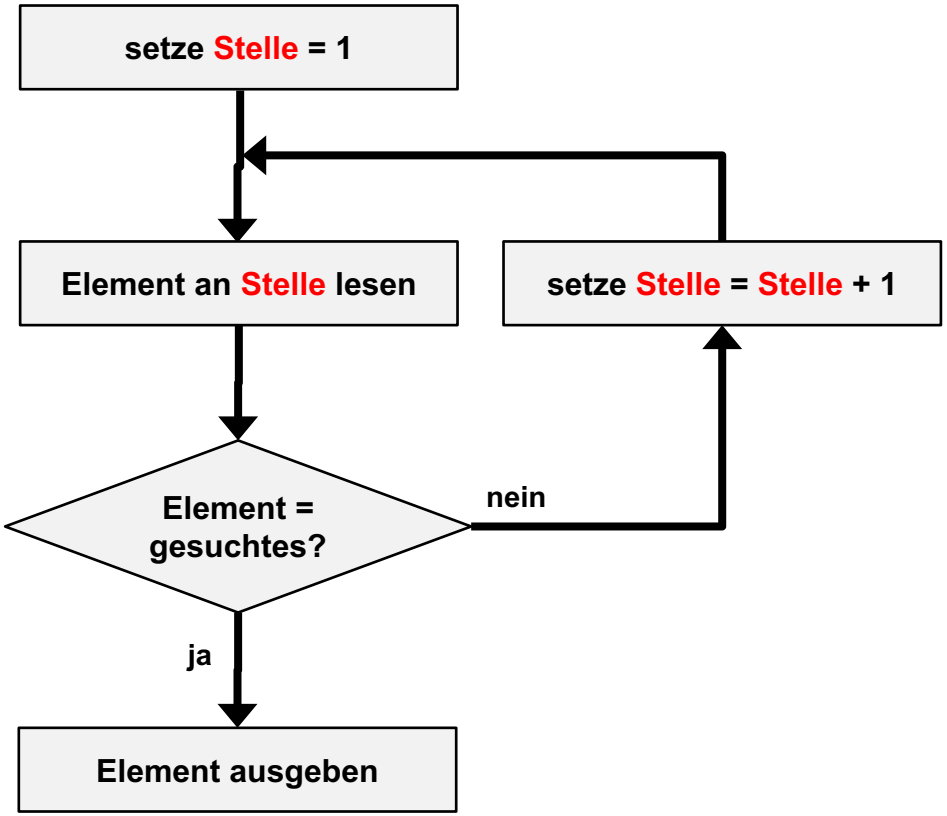
» können verschiedene Algorithmen zur Lösung desselben Problems vergleichen und beurteilen (z.B. lineare und binäre Suche, Sortierverfahren).

Beispiel: Lineare Suche vs. Binärsuche unplugged

Ein bestimmtes Element in einer grossen Datenmenge finden

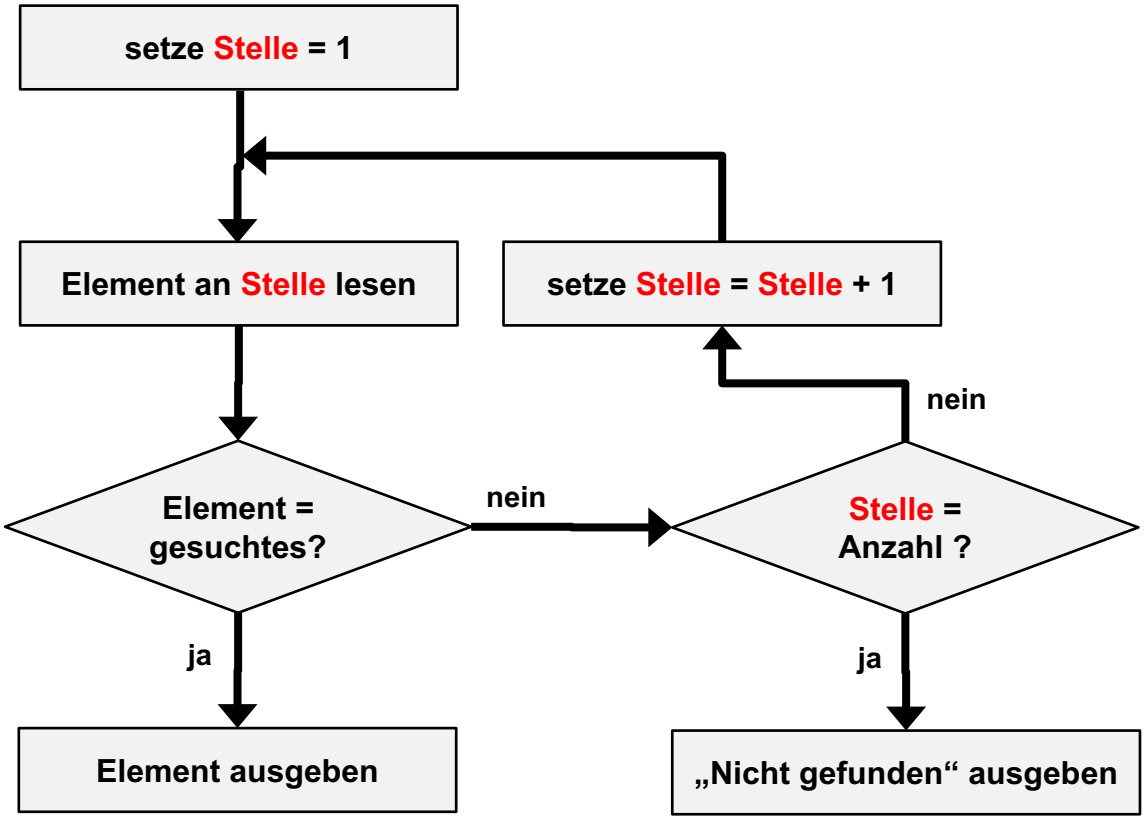
» können verschiedene Algorithmen zur Lösung desselben Problems vergleichen und beurteilen (z.B. lineare und binäre Suche, Sortierverfahren).

Lineare Suche als Flussdiagramm

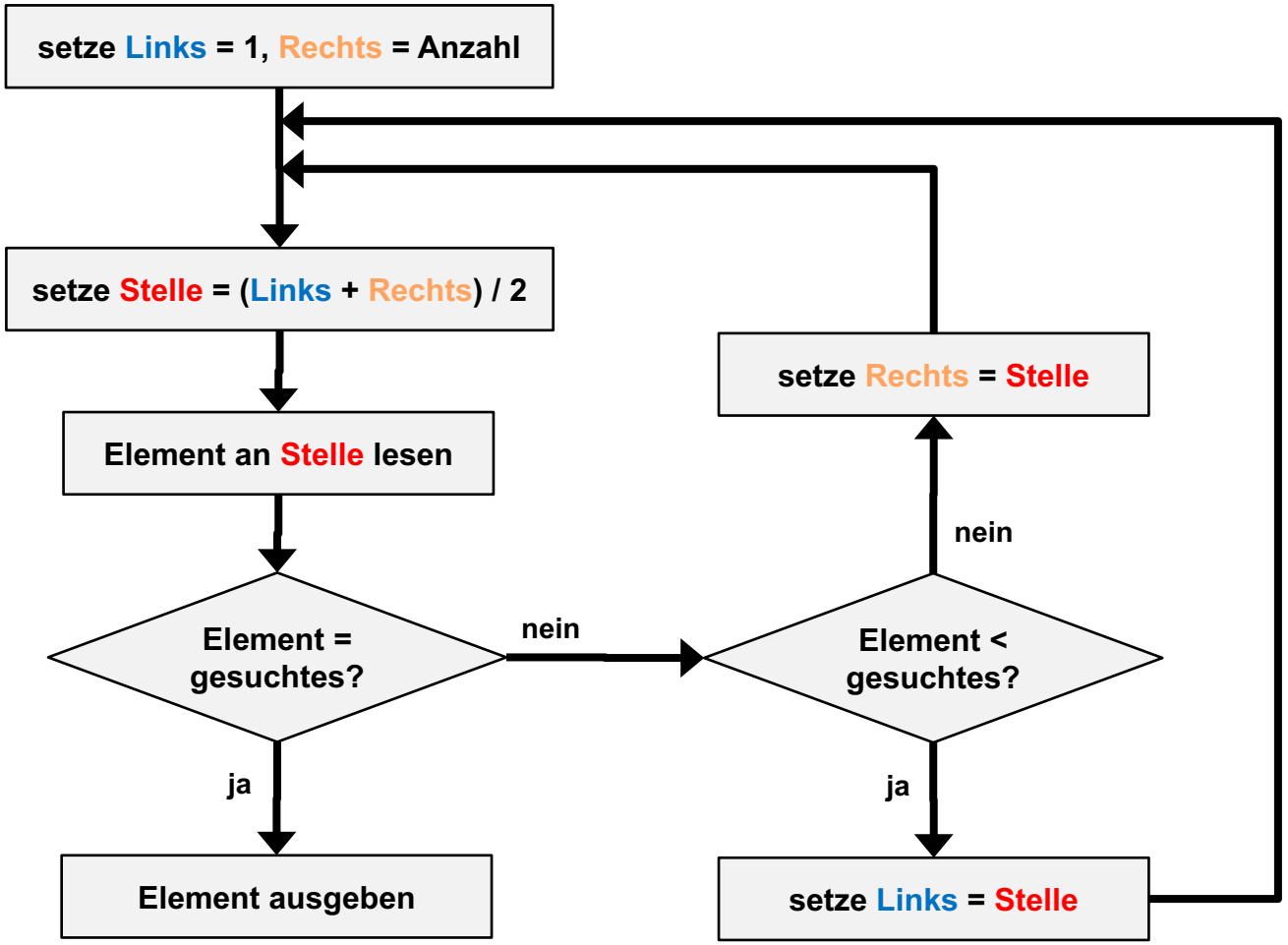


Gibt es bei diesem Algorithmus ein Problem?

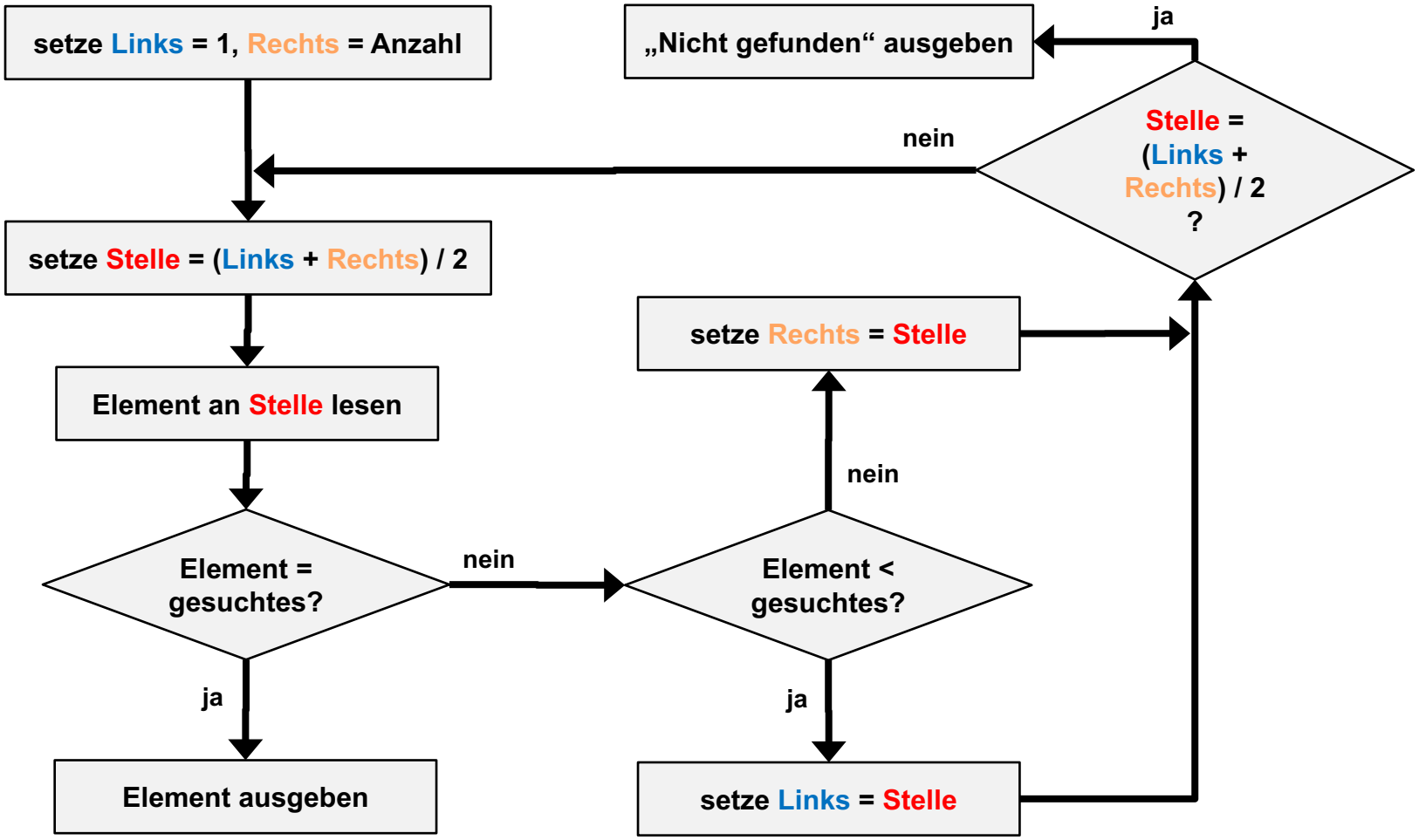
Lineare Suche als Flussdiagramm



Binärsuche als Flussdiagramm



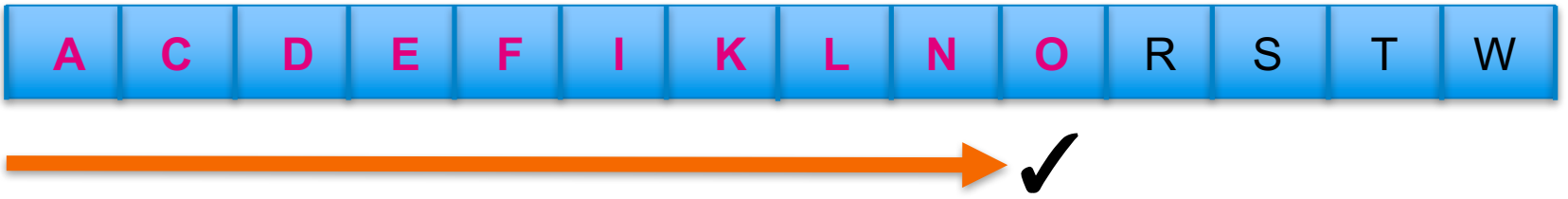
Binärsuche als Flussdiagramm



Beispiel – lineare Suche

Bei der **linearen Suche** beginnt man am Anfang der Liste und schaut jedes Element an, bis man das gesuchte gefunden hat.

Beispiel Such nach dem Eintrag „O“



10 Einträge wurden angeschaut und verglichen

Eine Liste mit 10‘000 Einträgen:

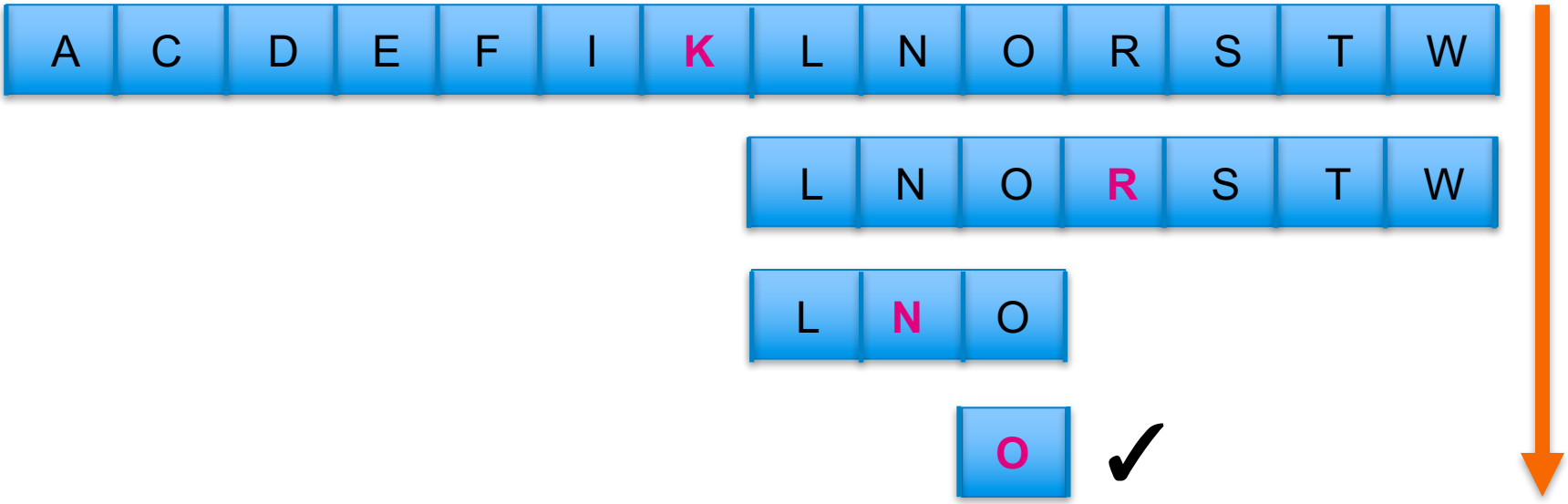
- Wie viele Einträge müssten wir im schlimmsten Fall anschauen? (alle)
- Wie viele Einträge müssten wir im besten Fall anschauen? (1)
- Wie viele Einträge müssten wir durchschnittlich anschauen? (5000)

Beispiel – Binärsuche

Bei der **Binärsuche** teilt man die vorsortierte Liste immer in zwei Hälften.

→ binäre Entscheidung, ob man mit der einen oder anderen Hälfte vorsetzt.

Beispiel Such nach dem Eintrag „O“

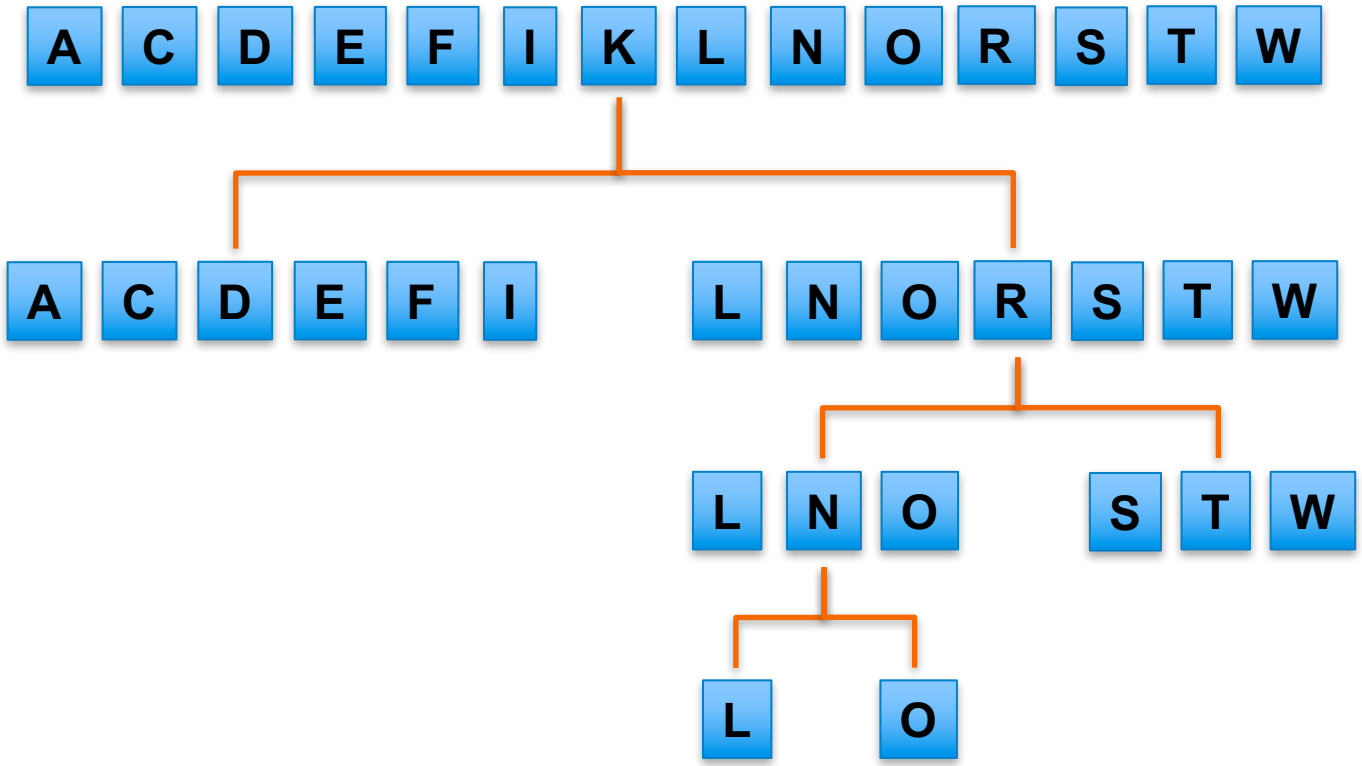


nur 4 Einträge wurden angeschaut

Eine Liste mit 10‘000 Einträgen?

schlimmsten Fall → nächsthöhere Zweierpotenz 2^**14** = 16384, damit **14** Einträge

Binärsuche



2^n	n
1	0
2	1
4	2
8	3
16	4
32	5
64	6
128	7
256	8
512	9
1024	10
2048	11
4096	12
8192	13
16384	**14**
32768	15
65536	16
131072	17

Eine Liste mit 10‘000 Einträgen?

schlimmsten Fall → nächsthöhere Zweierpotenz 2^**14** = 16384, damit **14** Einträge

→ Beispiel mit 100’000 Einträge wiederholen

lineare Suche vs. Binärsuche

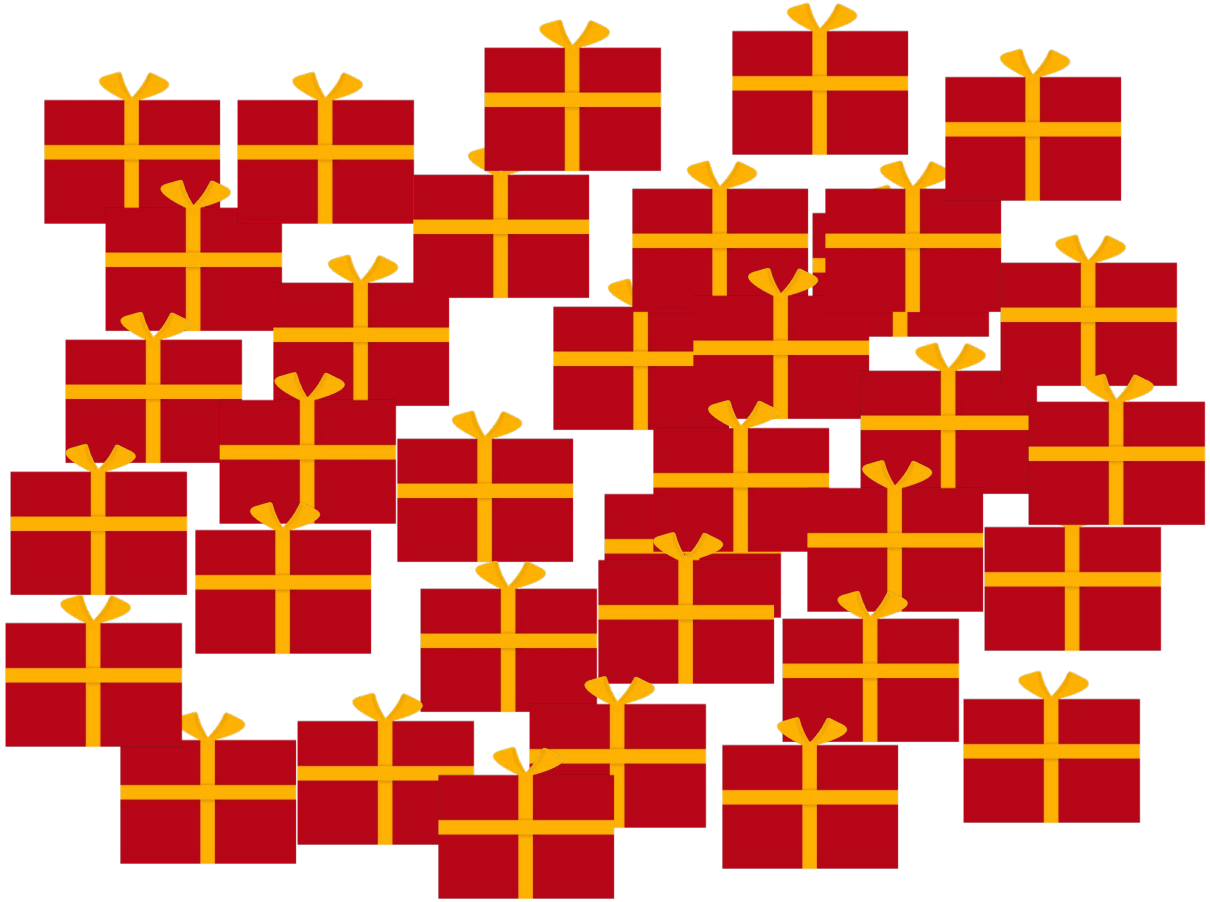
Erkenntnisse:

- Obwohl beide Such-Algorithmen genau das gleiche Ergebnis liefern, sind sie unterschiedlich schnell → effizient
- Als Nutzer sehen wir einem Programm in der Regel nicht an, welche Algorithmen eingesetzt werden, wir sehen nur das gefundene Ergebnis.
- ABER: Damit Binärsuche funktioniert benötigt man eine sortierte Liste!

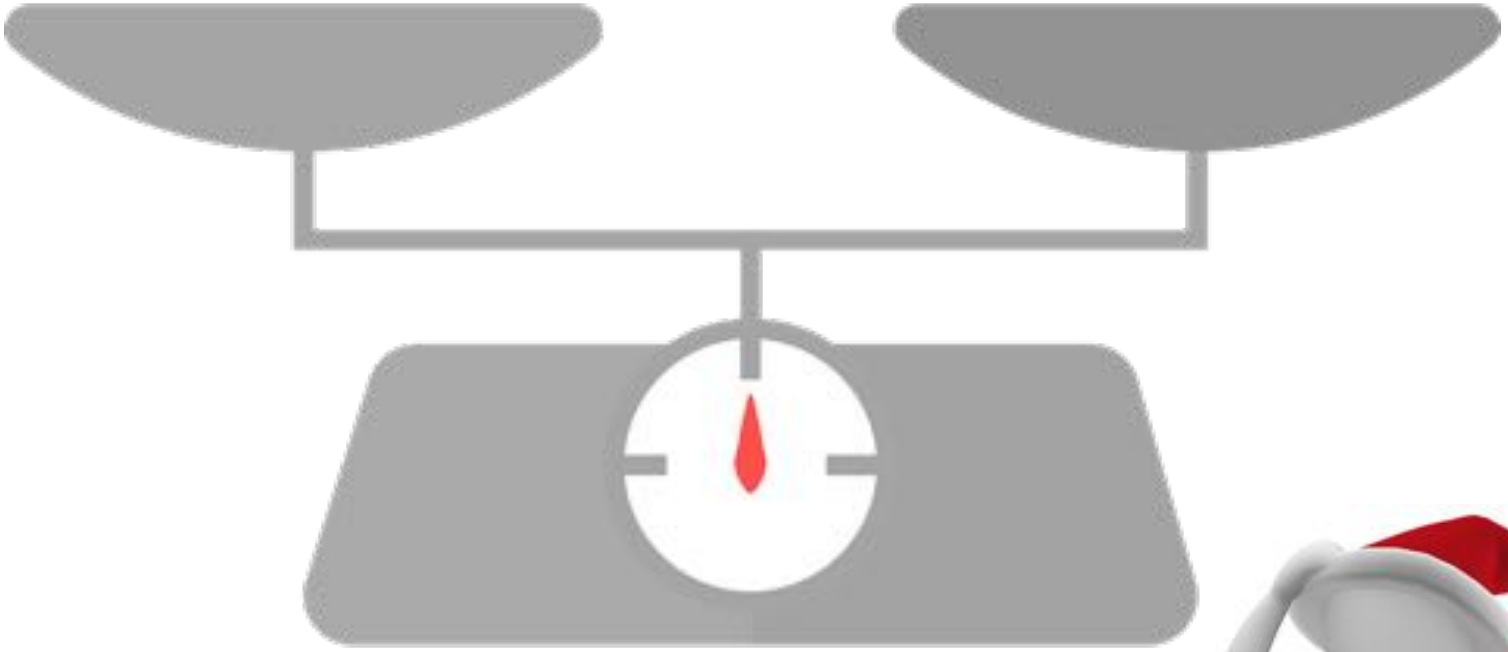
Such-Algorithmen können auch in anderen Situationen helfen →

Suchalgorithmen: Ein bestimmtes Element in einer grossen Datenmenge finden

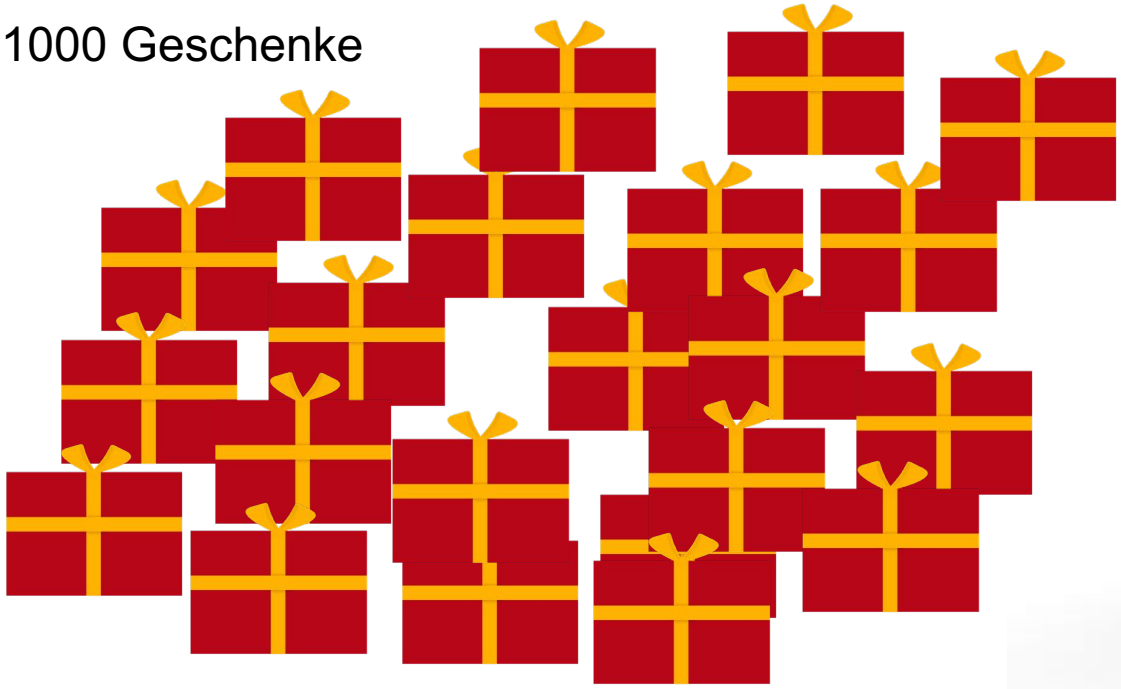
1000x  mit dem gleichen Spielzeugauto  darin



Oh, wo kommt denn der her?
Der fehlt jetzt in einem Geschenk!
Aber in welchem ???

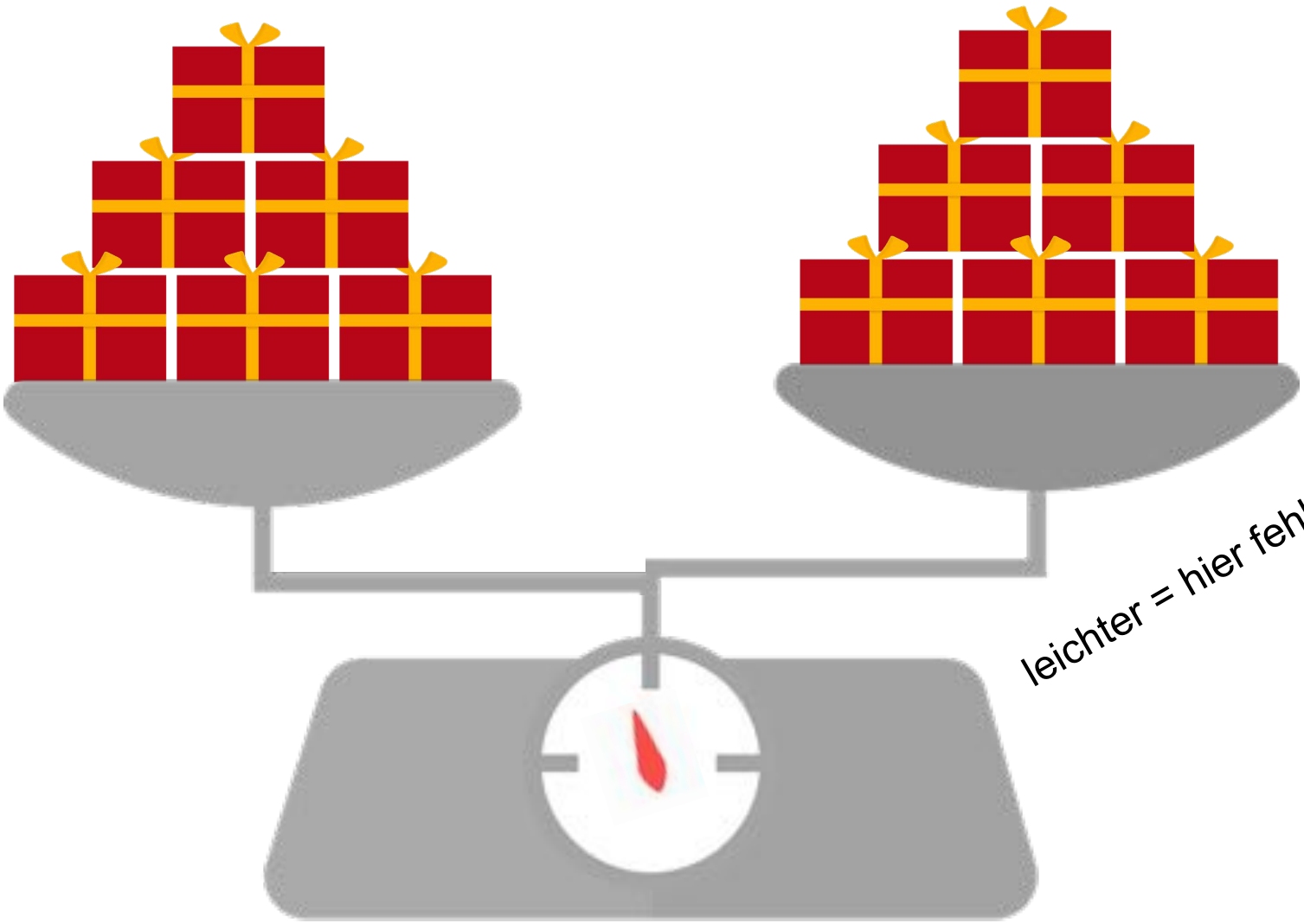


1000 Geschenke



500 Geschenke

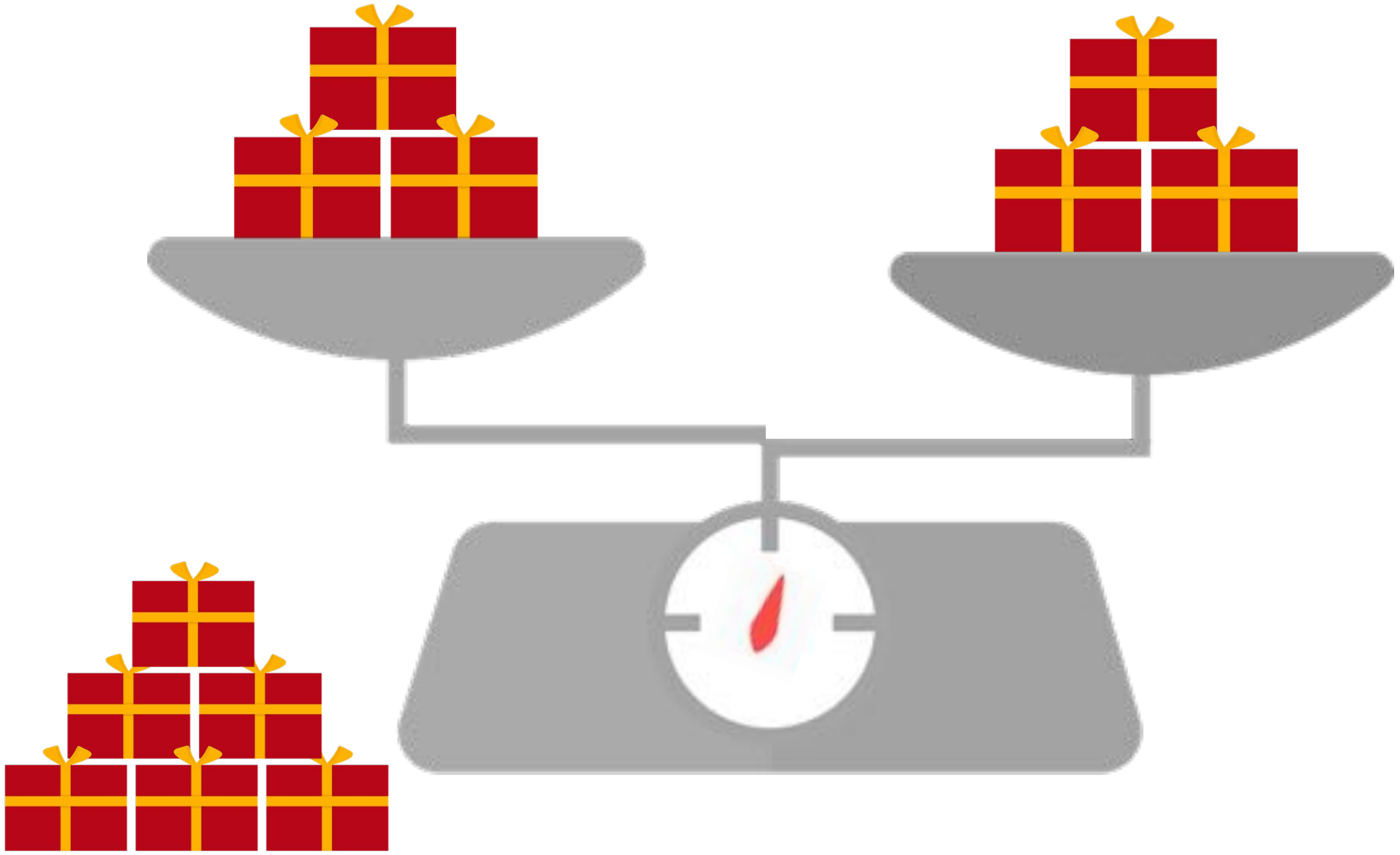
500 Geschenke



leichter = hier fehlt etwas

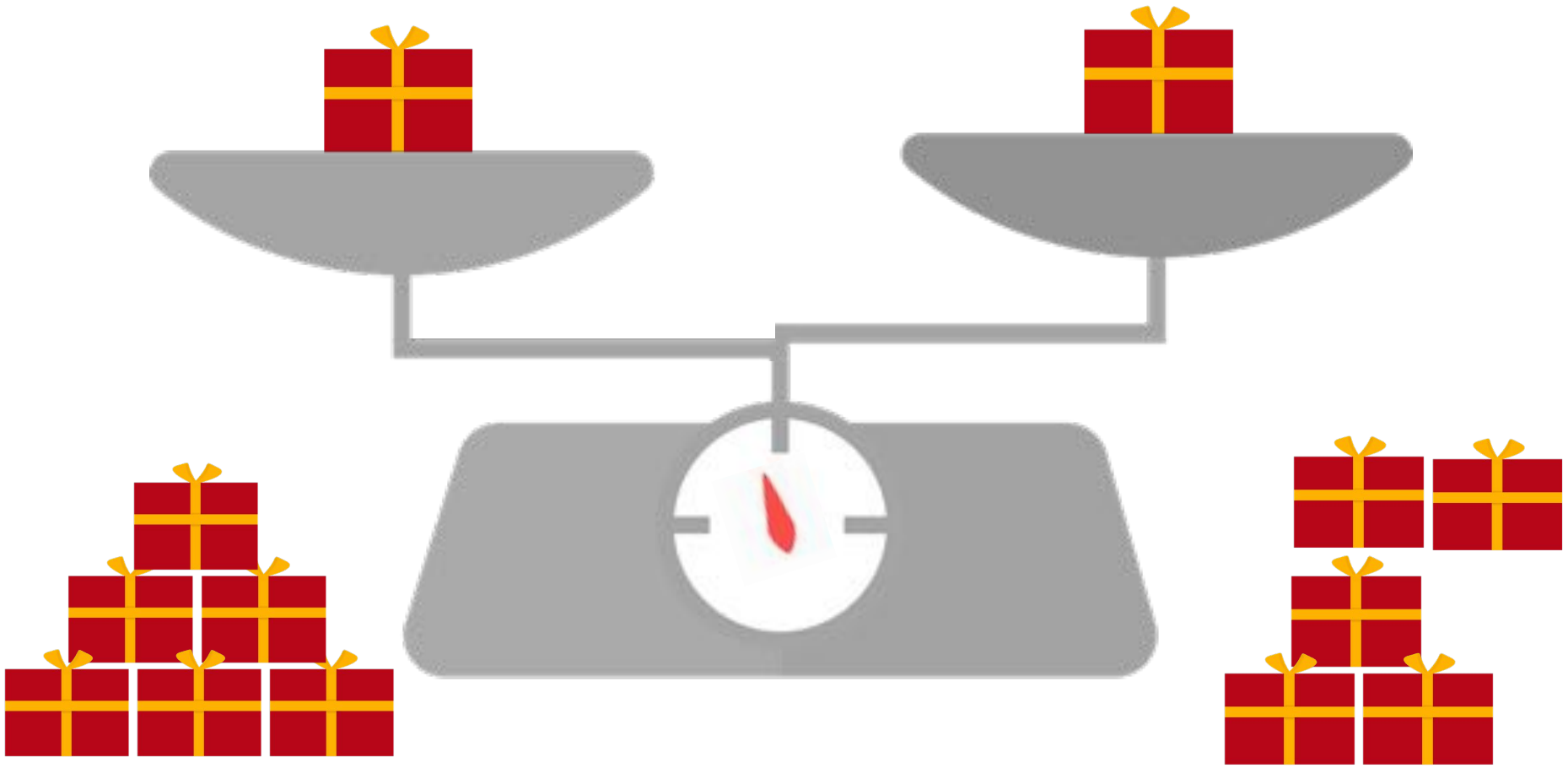
250 Geschenke

250 Geschenke



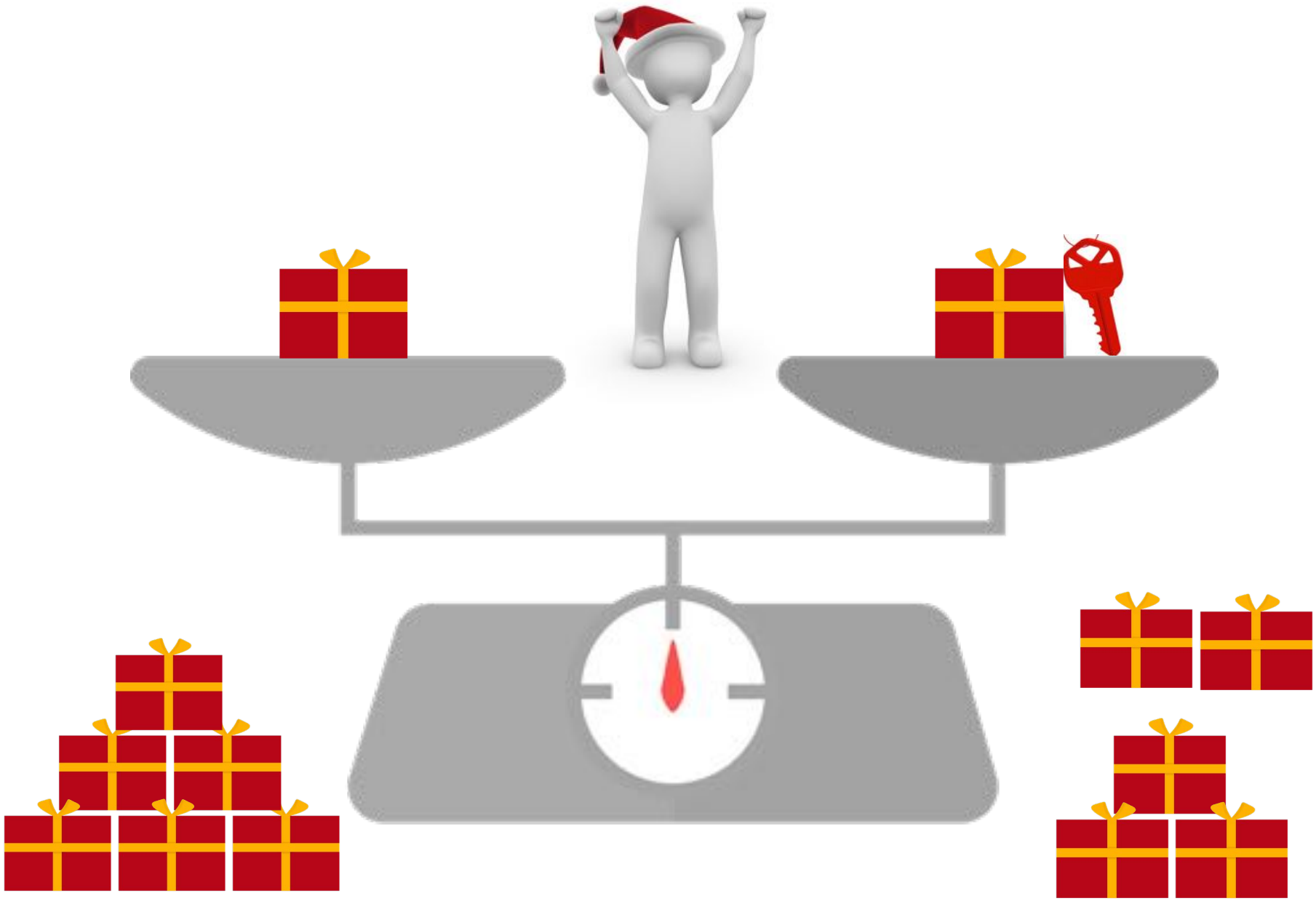
1 Geschenk

1 Geschenk



1 Geschenk

1 Geschenk

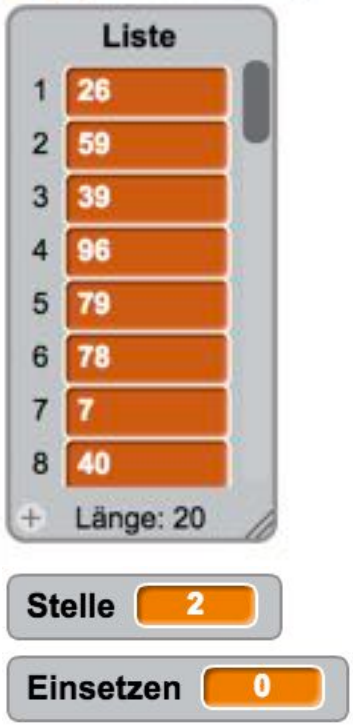


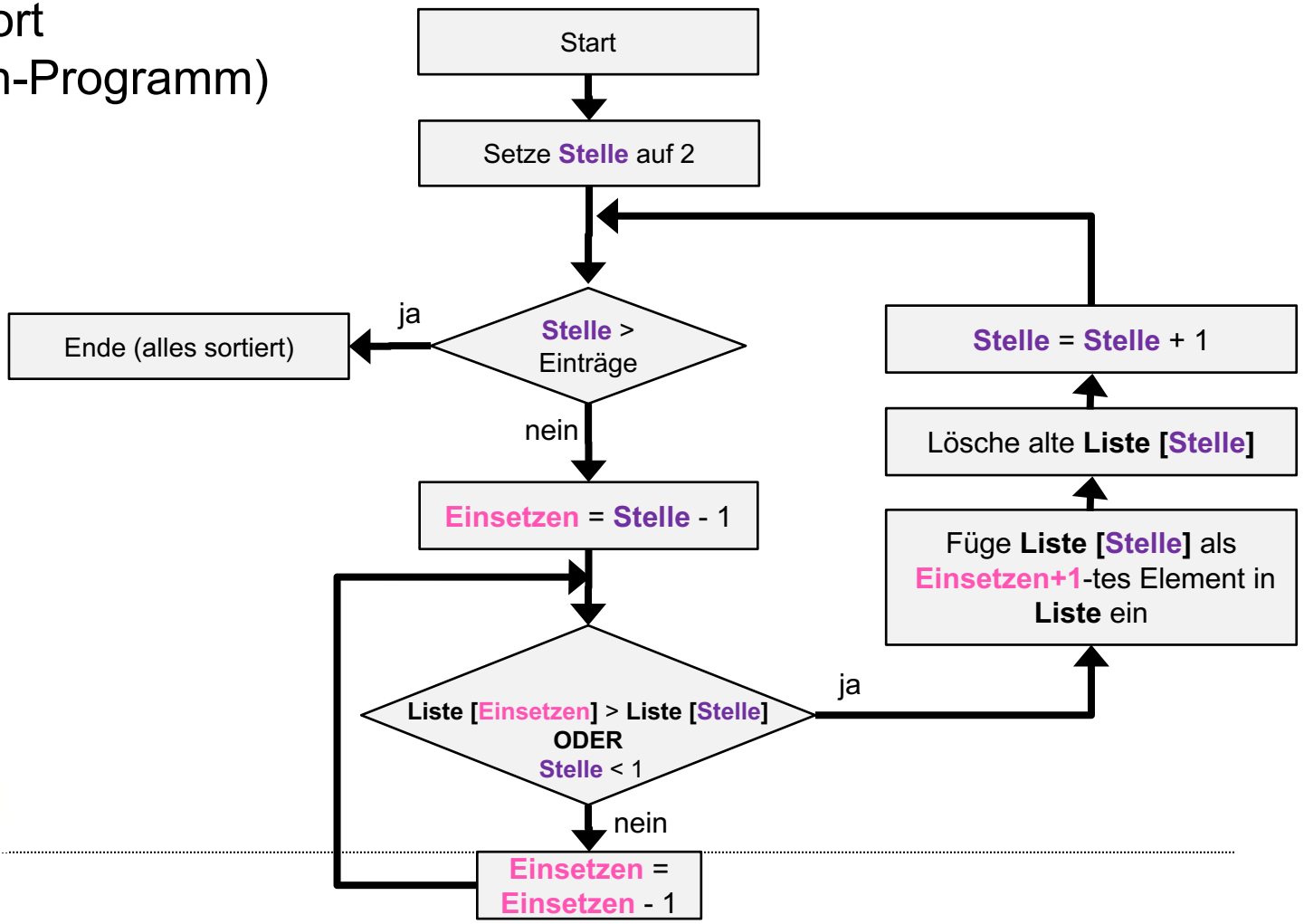
Damit die Binärsuche funktioniert benötigen wir Sortieralgorithmen.

Algorithmen Vergleichen - Sortieralgorithmen

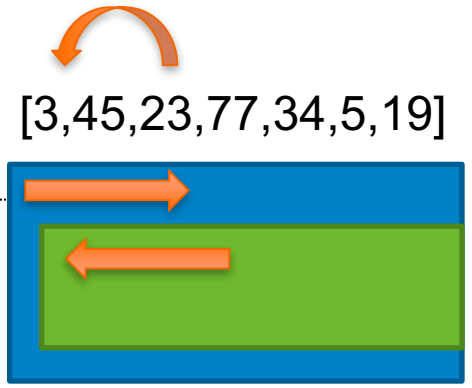
» können verschiedene Algorithmen zur Lösung desselben Problems vergleichen und beurteilen (z.B. lineare und binäre Suche, Sortierverfahren).

INSERTION-Sort
(kurzes Scratch-Programm)





Sortieren in Scratch – INSERTION Sort



In Worten:

- Schritt 1:
  Wir vergleichen den Wert an der aktuelle Stelle mit allen vorherigen, bis wir einen kleineren Wert gefunden haben, oder am Anfang angekommen sind.
- Schritt 2:
  Wurde ein kleineren Wert gefunden, wir das aktuelle Element in die Liste NACH dieser Stelle eingefügt. Anschliessend wird das aktuelle Element aus der Liste gelöscht (wir haben es ja an die andere Stelle „kopiert“).
- Schritt 3:
  die aktuelle Stelle wird um +1 erhöht und mit Schritt 1 fortgesetzt, solange bis wir am Ende der Liste angekommen sind.

Sortieren in Scratch – INSERTION Sort

Empfehlung:
Sortieren und Suchen unplugged.
nicht nachprogrammieren!

Dieses Verfahren umzusetzen ist eventuell nicht zielstufengerecht. Man könnte es als vorgegebenen Block oder Scratch-Programm bereitstellen.
Scratch ist mit seiner Darstellung wird mit wachsender Komplexität schwieriger zu lesen und zu verstehen.

Sortieren und Finden – unplugged
Verkehrshaus (iFactory)

» können verschiedene Algorithmen zur Lösung desselben Problems vergleichen und beurteilen (z.B. lineare und binäre Suche, Sortierverfahren).