

# PROGRAMMING BASICS

Dozent: Dr. Andreas Jäger

# LOGIK

Dozent: Dr. Andreas Jäger

# Übersicht

- Einführung
- Programmierparadigmen
- Prozedurale Programmierung
- Datentypen
- Operationen Funktionen
- Modellierung

# EINFÜHRUNG

Dozent: Dr. Andreas Jäger

# Einführung

Programmierung:

Spezielle Lösung eines Problems auf einer automatisierbaren Maschine

Es müssen Befehle, die der Automat interpretieren kann, in entsprechender Reihenfolge aneinander gereiht werden.

→ Verarbeitungsvorschrift / Algorithmus

# EVA

Dozent: Dr. Andreas Jäger

# EVA Prinzip

Wie ist ein Programm aufgebaut?

Das EVA Prinzip

Alle elektronischen Datenverarbeitungen kann man in folgende 3 Schritte unterteilen:



# Daten

Daten:

Eingabedaten(Input):

Ausgabedaten(Output):



# Daten

Daten:

keine einheitliche Definition!

Daten als Zeichen (oder Symbole) definiert, die Informationen darstellen

Eingabedaten(Input):

Daten die in das System eingelesen werden

Ausgabedaten(Output):

Daten die aus dem System ausgegeben werden

# Übung

Analysieren Sie im Hinblick auf das EVA-Prinzip die Tätigkeiten, die ein Mensch bzw. ein Computer durchführen muss, um die Hypotenuse  $c$  eines rechtwinkligen Dreiecks aus den beiden anderen Seitenlängen  $a$  und  $b$  zu berechnen.

- a) Welche Eingaben sind nötig?
- b) Welche Verarbeitungsschritte?
- c) Wie sollte die Ausgabe gestaltet werden?

# ALGORITHMEN

Dozent: Dr. Andreas Jäger

# Algorithmen

Ein Algorithmus ist eine eindeutige und endliche Beschreibung eines allgemeinen und endlichen Verfahrens zur Ermittlung gesuchter Größen aus gegebenen Größen.

# Übung

Diskutieren Sie in Kleingruppen über Beispiele für Algorithmen, die nichts mit Informatik zutun haben?

Nehmen Sie dazu die Def. einen Algorithmus zu Hilfe.

Jede Gruppe hat 10min ein Bsp. zu finden.

- Bedienungsanleitungen (Bsp. Für elektronische Geräte)
- Reparaturanleitungen (Für Fahrzeuge)
- Kochrezepte (Backen)

Was unterscheidet diese Bsp. von einem Algorithmus in der Informatik?!

→ Meist unvollständig oder weniger präzise

# Algorithmen

Ein Alg. muss vollständig und präzise sein!

warum?

Weil ein Computer nicht selbstständig entscheiden kann.

# Programme

Generelle Fragestellungen zum erstellen eines Programms:

Was soll das Programm können?

✓ Lösung für welches Problem!

Gibt es vorhandene Lösungswege?

✓ Finde Algorithmus für das Problem! Nutze: Erfahrung, Intuition, Ideen, Kreativität, Transfer

Falls dieser Ansatz nicht erfolgreich ist:

Kann man das Problem in Teile zerlegen? Divide and Conquer! Gibt es für die einzelnen Teile des Problems einen passenden Lösungsweg?



# BASICS

Dozent: Dr. Andreas Jäger

# Programmablaufplan

Wir lernen ein simples Werkzeug zur Programm-modellierung kennen.

Untermenge der Flussdiagramme (ISO 5807, DIN 66001)

Erlaubt uns die Modellierung von:

- Anweisung

- Verzweigung

- Ein-/Ausgabe

# Programmablaufplan

Vorteile:

    Simpel.

    Genügt um Grundlegende Kontrollstrukturen zu verdeutlichen.

Nachteile:

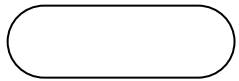
    Keine Komplexen Beziehungen Modellierbar.

    Eingeschränktes Toolset.

# Symbole



**Flusspfeil:** Der Pfad des Ablaufs wird durch Pfeile dargestellt. Bei der Visualisierung eines Programmablaufs handelt es sich somit um einen gerichteten Graphen. Pfeilspitzen sind daher unverzichtbar.



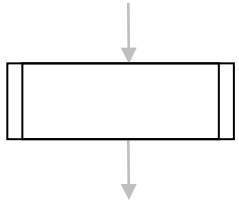
**Modulgrenze:** Start bzw. Ende eines Moduls wird durch eine Ellipse gekennzeichnet. Im Start findet man eine Angabe über Art des Moduls (z.B. *Flächenberechnung*). Im Ende die Ausgangsparameter des Moduls (z.B. Rückgabewerte).

Für die Programmierung versuchen wir nur ein einziges Ende pro Modul zu definieren.

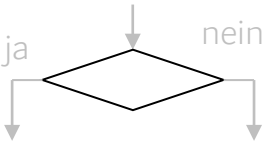


**Prozess:** Abarbeitungsschritt in einer Sequenz von Prozessen. In der Programmierung entspricht dies einer Anweisung, die den Zustand des Programmes (Variablen) durch Berechnung/Zuweisung ändert. Anweisung im Rechteck genannt.

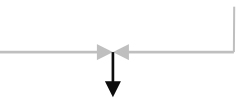
# Symbole



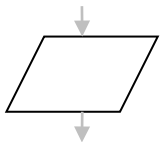
**Vordefinierter Prozess:** Ausführen eines Submoduls. (Optional, kann auch durch „Prozess“ erstellt werden. Name des Submoduls im inneren Rechteck angeben.



**Bedingungsprüfung & Verzweigen eines Pfades:** Das Raute-Symbol beschreibt die Möglichkeit den Pfad in zwei Richtungen aufzuteilen. Pfade können auch wieder zurückführen (→ Schleifen).

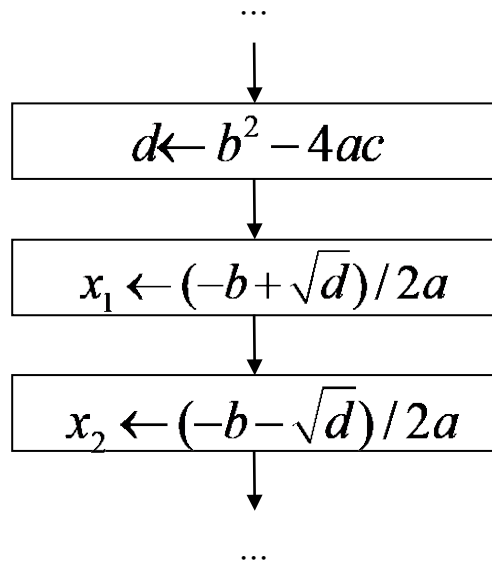


**Pfade zusammenführen:** Verzweigte Abläufe (durch Raute) müssen auch wieder an einer Stelle zusammengeführt werden.



**Daten:** Ein- und Ausgaben werden durch das Parallelogramm dargestellt. Es können sowohl Texte als auch Werte aus Variablen ausgegeben werden (mehr beim Programmieren).

# Prozesse - Anmerkung

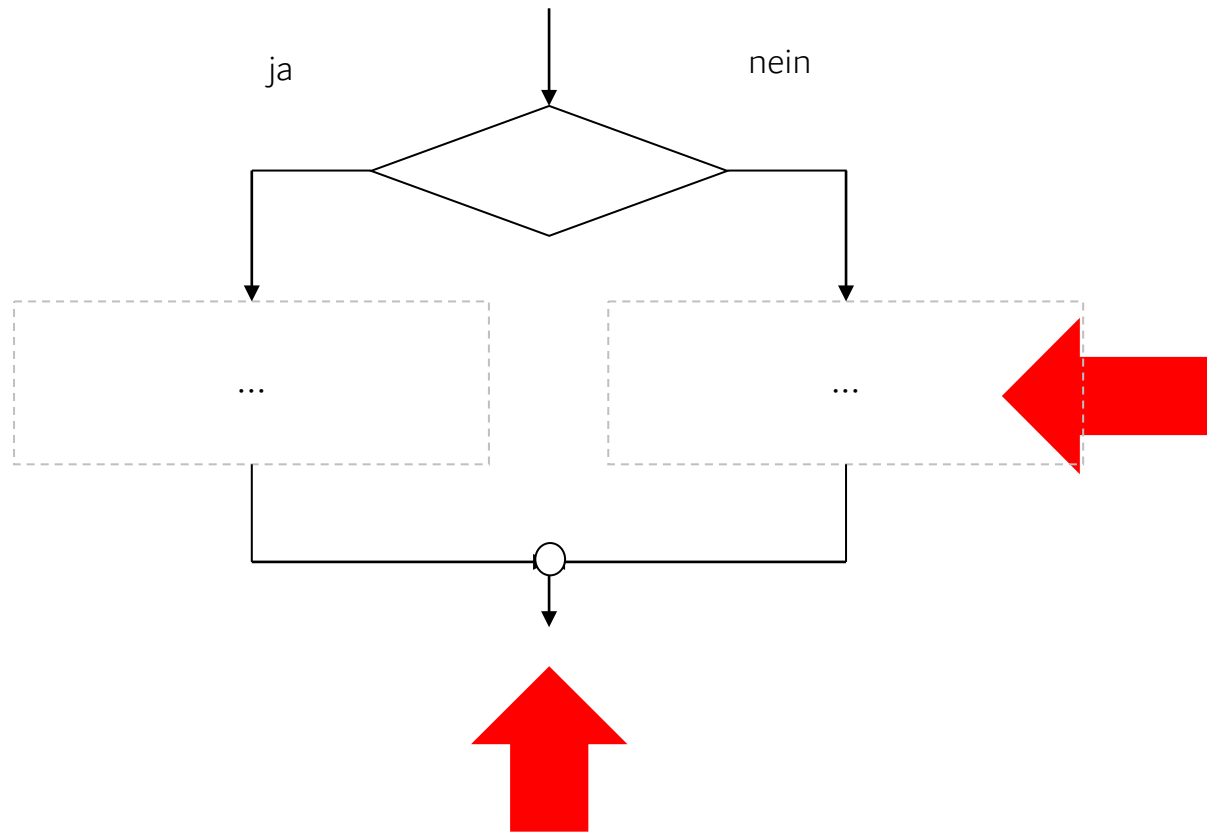


**Wichtig:** Wir arbeiten mit Prozess-Schritten (Anweisungen!) und nicht mit mathematischen Definitionen bzw. Sätzen o.ä.!

D.h. alle Anweisungen werden Schritt für Schritt in diesem Moment durchgeführt und mit dem entsprechenden vorhandenen Wert (Zustand) berechnet!

# Selektion

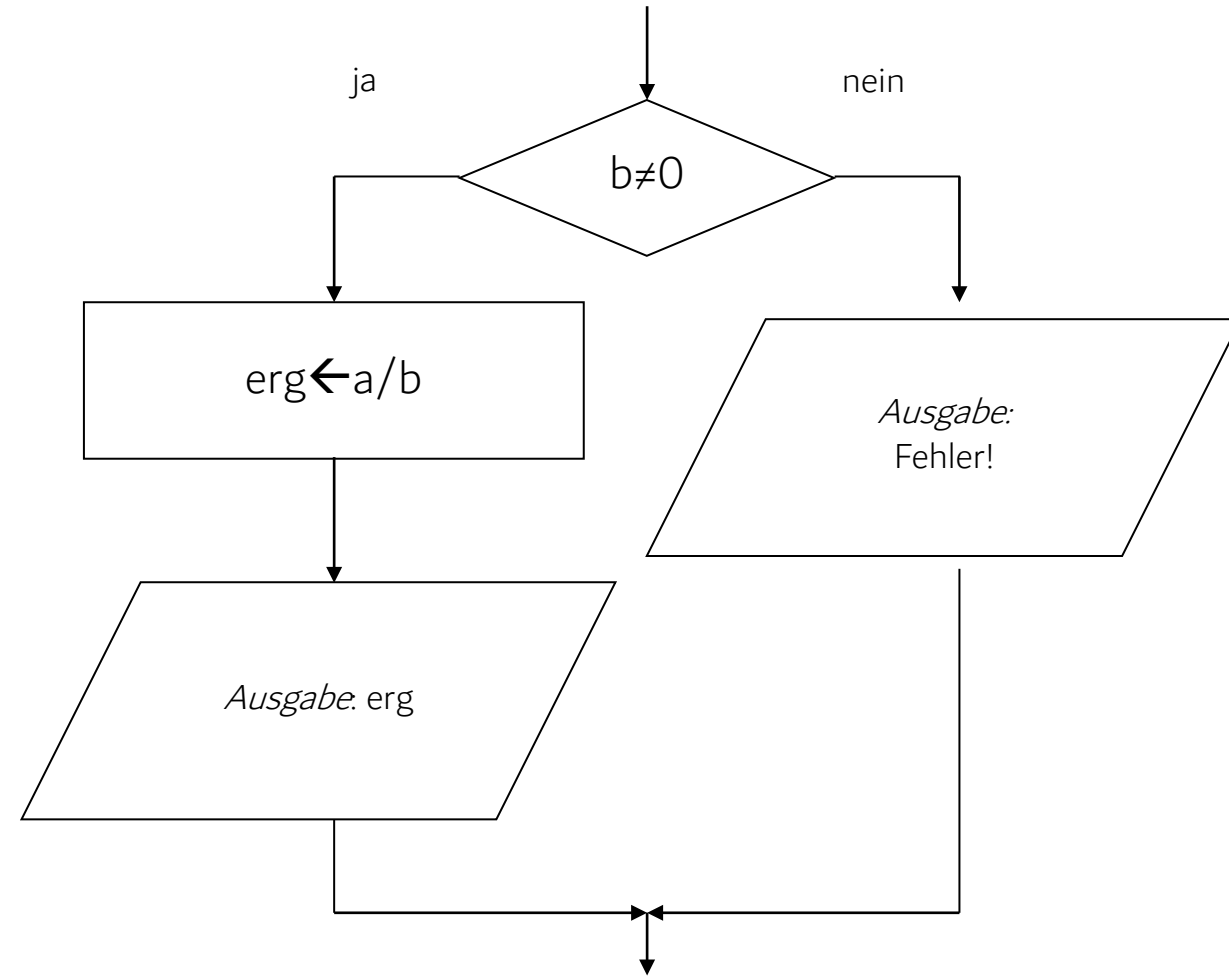
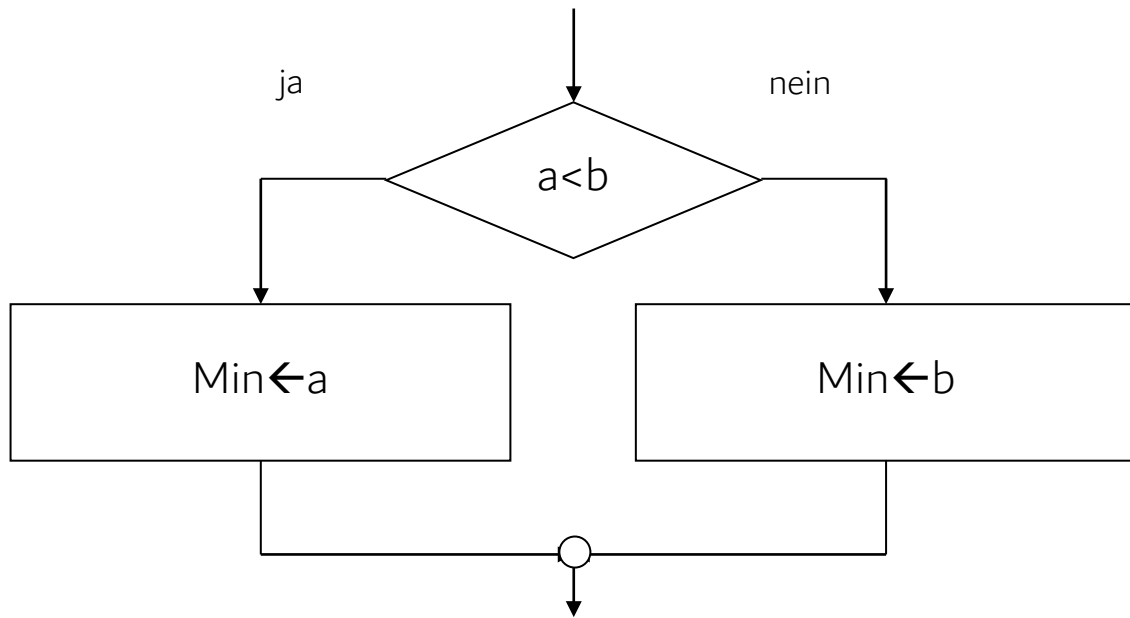
Symbol



Block mit weiteren  
Kontrollstrukturen

Zusammenführung in  
geschachtelter Form!

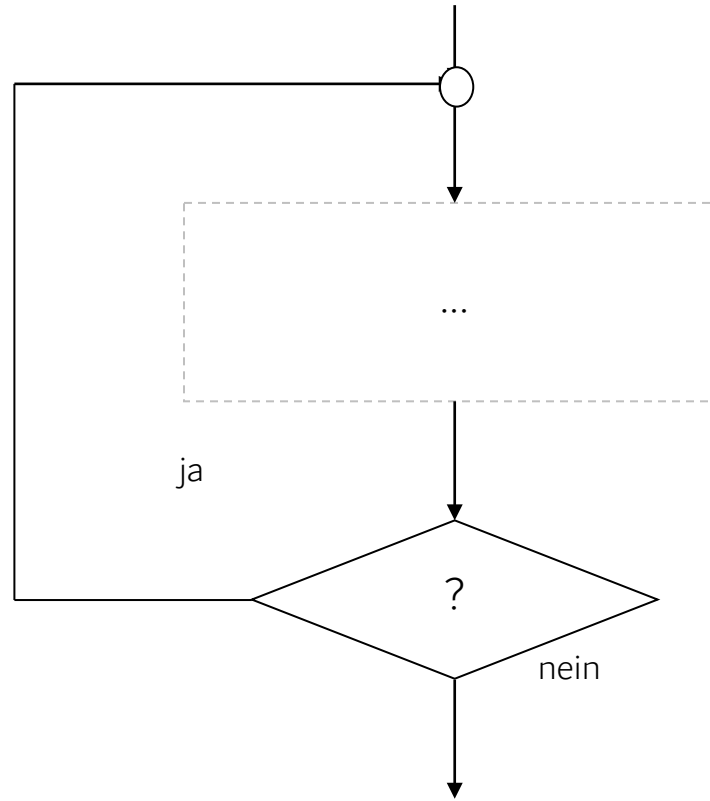
# Selektion



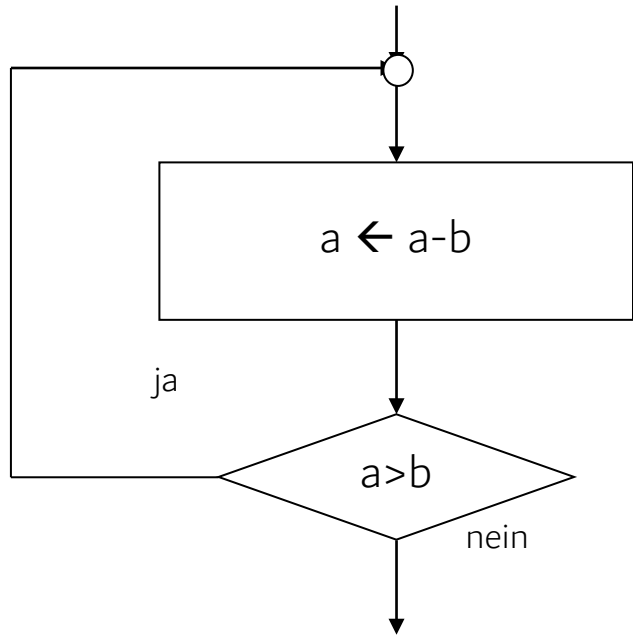


# Iteration (Schleife)

Symbol

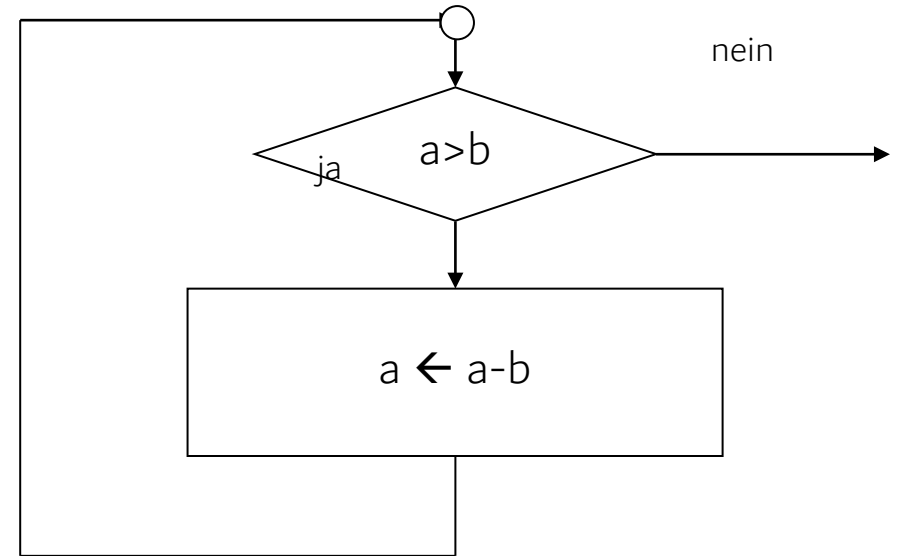


# Iteration (Schleife)




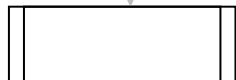
Fußgesteuert

Kopfgesteuert



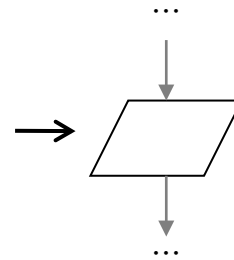
# Abweichungen von der Norm

Folgende Anpassungen nehmen wir für die Vorlesung vor:

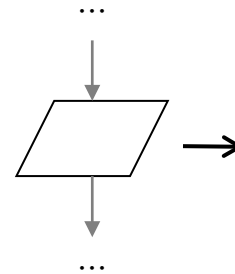
- Zusammenführung kann auch  verwendet werden
- Verzicht von Symbol für „vordefinierten Prozess“ 
- Wir versuchen möglichst nur ein Ende pro Modul zu definieren.
- Wir versuchen im „Sprungteil“ von Schleifen keine weiteren Aktionen durchzuführen.

# Abweichungen von der Norm

Eingabe von  
Daten



Ausgabe von  
Daten



# Übung

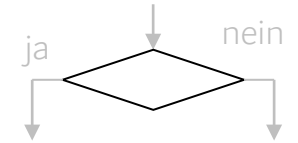
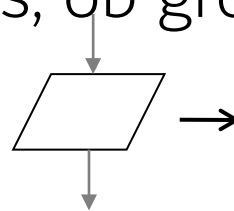
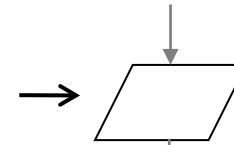
Welches Symbol wird verwendet:

- a. Addition von a und b
- b. Eingabe eines Namens
- c. Überprüfen eines Wertes, ob größer als ein anderer Wert
- d. Ausgabe eines Textes

# Übung

Welches Symbol wird verwendet:

- a. Addition von a und b
- b. Eingabe eines Namens
- c. Überprüfen eines Wertes, ob größer als ein anderer Wert
- d. Ausgabe eines Textes

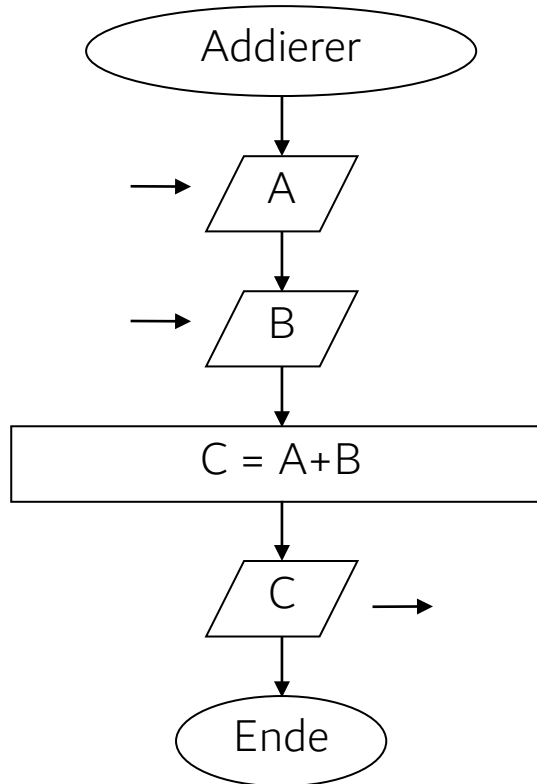


# Übung

- a) Modellieren Sie einen Taschenrechner, der 2 Zahlen addiert und das Ergebnis ausgibt.

# Übung

- Modellieren Sie einen Taschenrechner, der 2 Zahlen addiert und das Ergebnis ausgibt.



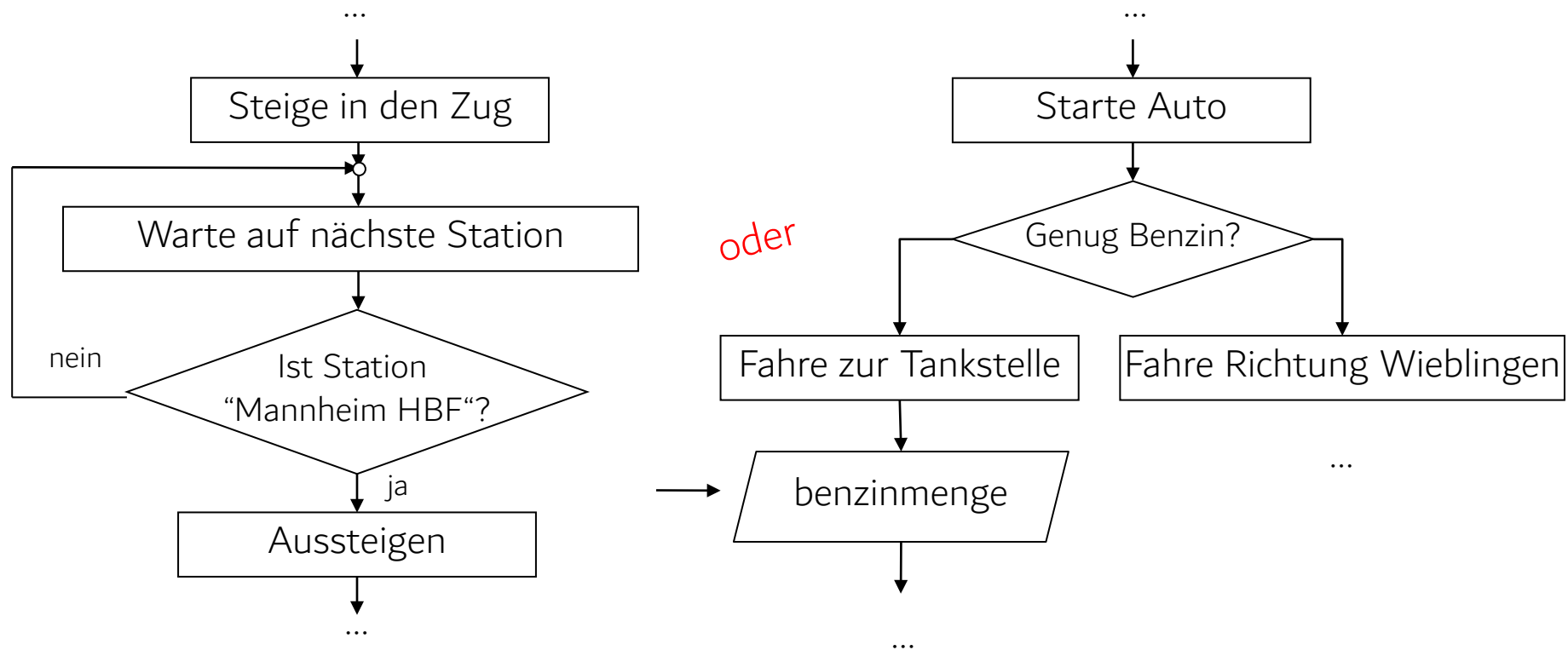


# Übung

- Erweitern Sie den Taschenrechner auf 4 Grundrechenarten + \* - /

# Übung

Beschreiben Sie Ihren Weg zur Hochschule mit Hilfe eines Programmablaufplans





VIELEN DANK  
FÜR  
IHRE AUFMERKSAMKEIT!

