Fachhochschule Jena
University of Applied Sciences Jena

Informatik I

Kapitel 1: Einführung

Burkart Voss

Das sollten Sie am Ende können!!!

- **Schreiben Sie eine C-Funktion teile(i,j,k,l), die die übergebene Zahl i durch j dividiert und in k das Ergebnis sowie in l den Divisionsrest zurückliefert. Alle Zahlen sollen ganze Zahlen sein. Die Funktion teile soll eine Division durch 0 abfangen und in diesem Fall den Rückgabewert 0 liefern, sonst den Rückgabewert 1.**

- **Erarbeiten Sie eine Funktion mit folgendem Interface:**

```
struct formular *sortiert_einfügen(struct formular *kartei,/
struct formular *anfang);
```

Die Funktion soll wie folgt verwendet werden:

Es soll davon ausgegangen werden, dass eine sortierte verkettete Liste existiert, deren Elemente vom Typ `struct formular` sind und deren erstes Element im zweiten übergebenen Parameter `anfang` der Funktion übergeben wird. Es soll die Adresse des ersten Elements der Liste zurückgegeben werden, nachdem das als erster Parameter übergebene Element an der richtigen Stelle eingefügt wurde.

- **Turnus:**
  - **Wochen 1-5:**
    - **2 Doppelstunden Vorlesung**
  - **Wochen 6-15:**
    - **Jede Woche 3x45 min Praktikum**
    - **14 tägig Vorlesung**
- **Bewertung:**
  - **Es sind eine STUDIENLEISTUNG und eine PRÜFUNGSLEISTUNG zu erbringen**

Studienleistung

Fachhochschule Jena
University of Applied Sciences Jena

- **Kurztests**
  - **Am ANFANG ausgewählter Vorlesungen und Praktika wird ein bewerteter Kurztest geschrieben (insgesamt 6 über das Semester verteilt).**
- **Praktika**
  - **Gruppen zu max. 16 Studenten**
  - **Praktikum findet im Mikrorechnerlabor statt**
  - **In den Praktika sollen die Programmierfähigkeiten in C gefestigt werden.**
  - **Im letzten Praktikum wird eine Programmieraufgabe gestellt, die während dieses Praktikums selbstständig gelöst werden muss.**
- **Die Studienleistung wird erteilt, wenn**
  - **die Programmieraufgabe selbstständig gelöst wurde UND**
  - **mindestens 4 von 6 Kurztests bestanden wurden.**
  - **Die Kurztests können nicht wiederholt werden und werden nur zu den festgelegten Terminen angeboten.**

Modulprüfung

Fachhochschule Jena
University of Applied Sciences Jena

- **Schriftliche Klausurarbeit**
  - **Dauer: 90min**
  - **Hilfsmittel: keine**
  - **Relevanter Stoff: Der Stoff aus den Vorlesungen und Praktika**
  - **Wiederholungsmöglichkeit: nach 1 Semester**
  - **Achtung – Note ist Bestandteil des Abschlußzeugnisses!**

- **C von A bis Z von Jürgen Wolf**
  „Das umfassende Handbuch für Linux, Unix und Windows“ - 2., aktualisierte und erweiterte Auflage 2006

  Dieses Buch bietet Programmiereinsteigern und Studenten technischer Fächer einen umfassenden Einstieg in C. Auch für fortgeschrittene C-Programmierer ist das Buch eine ausgezeichnete Fundgrube.

  Nutzen Sie die HTML-Version des Buches zum Reinschnuppern oder als immer verfügbare Ergänzung zu Ihrem Buch.

- **Als „openbook“ kostenlos verfügbar unter:**
  http://www.pronix.de/pronix-4.html

Weitere Literaturempfehlungen

- Robert Klima und Siegfried Selberherr: “Programmieren in C”. Springer-Verlag Wien, 2003
- Jürgen Bayer: “Programmieren lernen – Anfangen Anwenden Verstehen”. Addison-Wesley 2003
- Andrew Hunt und David Thomas: “Der pragmatische Programmierer” Hanser Verlag, 2003
- Mike Banahan, Declan Brady and Mark Doran: “The C-Book”. Addison Wesley, 1991 (freely available as e-book at http://publications.gbdirect.co.uk/c_book/)
- Steve Oualline: “C – Elements of Style”. M&T books 1992. (freely available as e-book at http://www.computer-books.us/c_3.php)

www.et.fh-jena.de/voss

- **Hier werden die Vorlesungsfolien vor der Vorlesung veröffentlicht.**
  - **Bitte ausdrucken und zu der Vorlesung mitbringen, da dort Ergänzungen vorzunehmen sind!**
- **Hier werden die Aufgaben für die Praktika veröffentlicht.**
  - **Bitte VOR den Praktika herunterladen und sich damit beschäftigen!**

Informationsverarbeitendes System

- **Aufgabe: Ein bestimmtes Problem soll automatisch gelöst werden.**
  - **Für diese Lösung muß eine Maschine (System) gebaut werden. Maschine besteht aus:**
    - **Computer (Rechner, Rechenautomat) - Hardware**
      - kann alleine nichts - und
    - **Programm(e), das den Computer steuert und damit das Computerverhalten festlegt (Software)**

  spezielle Maschine = Computer + Programm

Wie vorgehen?

- **Problem verstehen! → Analyse**
- **Lösungsziel genau beschreiben → Spezifikation**
- **Wie kann Problem im Rechner abgebildet werden → Datenstrukturen**
- **Verfahren für das Erreichen der Lösung finden → Algorithmus**
- **Implementieren des Algorithmus → Programm**

- **Wie sage ich dem Computer, was ich eigentlich will →**
  - **Datenstrukturen modellieren die Realität im Computer**
  - **Algorithmen arbeiten auf den Datenstrukturen**

- **Algorithmisches Denken muss gelernt sein**
  - **GUT → man MUSS das nicht auswendig lernen**
  - **SCHLECHT → man KANN das nicht auswendig lernen**

→ ÜBEN!!!

Algorithmen und Programm

Fachhochschule Jena
University of Applied Sciences Jena

- **Algorithmen sind Handlungsanweisungen**
  - **Beispielsweise um Essen zu kochen**
  - **um ein Auto zu bauen**
  - **um sich an der FH einzuschreiben**
  - **um die jährliche Tilgung eines Kredits zu berechnen**
- **Elementare Aktionen werden als bekannt vorausgesetzt**
  - **rühren, erhitzen, Gemüse schneiden, …**
  - **Schraube eindrehen, Punkt schweißen, …**
  - **Formular ausfüllen, Geld überweisen, …**
  - **Multiplizieren, addieren, … - dem Verarbeiter bekannt!**
- **Ein Algorithmus beschreibt eindeutig**
  - **Reihenfolge**
  - **Bedingungen**

  der Ausführung dieser „elementaren“ Aktionen (der Schritte)

Ein Programm ist die Beschreibung eines Algorithmus in einer bestimmten Programmiersprache.

Schritte eines Algorithmus – Beispiele

- **Elementare Operation:**
  - **„Schneide Fleisch in kleine Würfel.“**
- **Sequentielle Ausführung:**
  - **„Bringe das Wasser zum Kochen, dann gib das Paket Nudeln hinein, schneide das Fleisch, dann das Gemüse.“**
- **Bedingte Ausführung:**
  - **„Wenn die Soße zu dünn ist, dann füge Mehl hinzu.“**
- **Wiederholte Ausführung (Schleife):**
  - **„Rühre, bis die Soße braun ist.“**
- **Teilverfahren (Unterprogramm):**
  - **„Bereite Soße nach Rezept auf Seite 43 (des Kochbuches).“**
- **Rekursion:**
  - **„Halbiere die Torte bzw. das Tortenstück in zwei gleich große Stücke. Falls noch nicht alle Gäste ein Stück haben, verfahre mit den einzelnen Stücken wieder genauso (bis alle Gäste ein Stück Torte haben).“**

Wie werden Algorithmen formuliert?

- **Formulierung muß dem Verarbeiter – dem Prozessor – verständlich sein,**
  - **muß präzise sein!**
- **Umgangssprache zur Beschreibung des Verfahrens,**
  - **z.B. Kochrezept, Bedienungsanleitung; oder**
- **Pseudocode:**
  - **Umgangssprache, die durch „verabredete“ Formulierungsformen und sogenannte „Schlüsselworte“ strukturiert ist, sich aber auch von Nichtprogrammierern lesen und verstehen läßt.**

    Die Schlüsselworte kennzeichnen den Verfahrensablauf.
- **Graphisch:**
  - **(Programm-) Ablaufplan oder**
  - **Struktogramm**
- **Programmtext (Quellcode)**

Wie viele Elemente braucht man für beliebige Algorithmen?

- **Sequenz**
- **Verzweigung**
- **Wiederholung/Schleife**

deutsche Schlüsselworte oder entsprechende englische

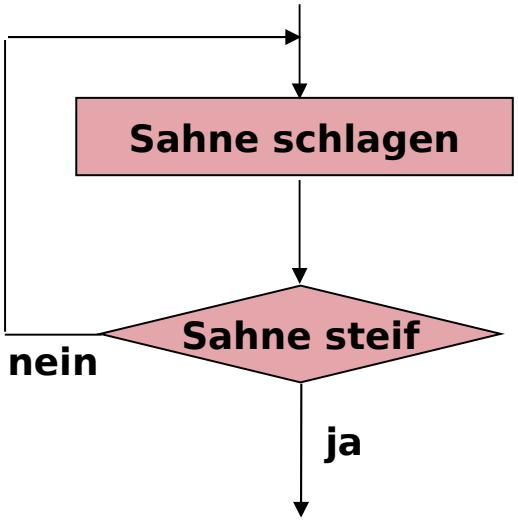
Deutsch	Englisch
wenn dann sonst	if then else
solange tue	while do
wiederhole bis	repeat until
tue bis	do while
wiederhole für	for … do oder for (…..)

Beispiel Wiederholung

...
Tue

die Sahne schlagen

bis **die Sahne steif ist**
...



Beispiel – Minimum von 2 Zahlen bestimmen

Start

A ← von Eingabe

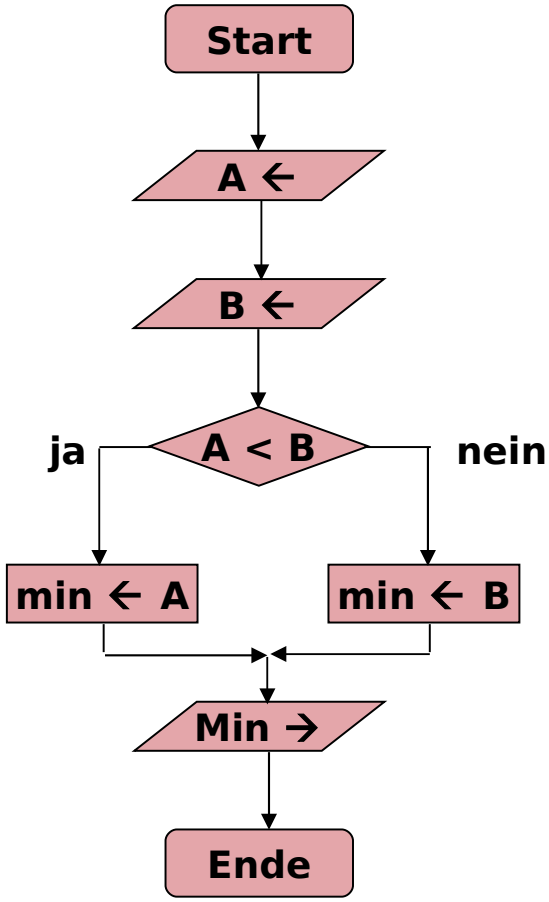
B ← von Eingabe

Wenn **A < B**

dann **min ← A**
sonst **min ← B**

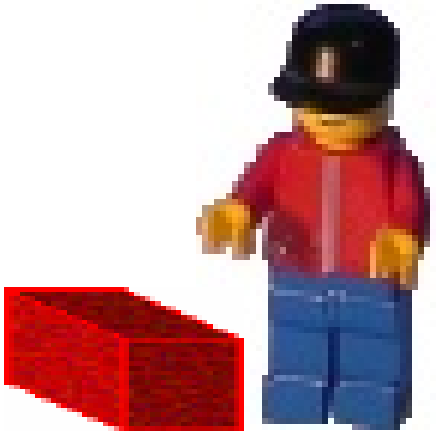
Ausgabe min

Ende.



Karol the Robot

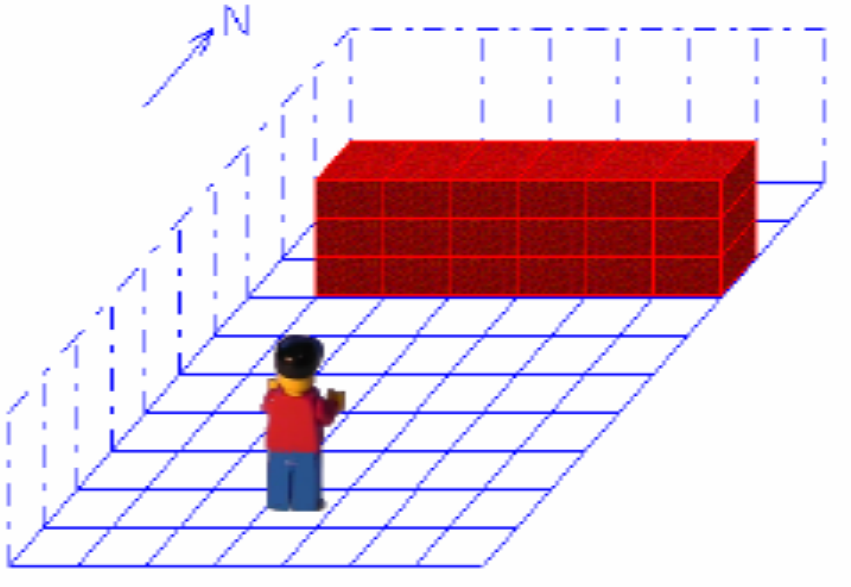
- **In der Welt von Karol gibt es:**
  - Ziegel, die Karol vor sich ablegen und aufsammeln kann
  - Marken, die Karol aufsammeln und ablegen kann
  - Und KAROL



- **Karol hat Funktionen, mit denen**
  - er aktiv werden kann:
    - Mache einen Schritt vorwärts! — **Schritt**
    - Drehe dich um 90° nach links! — **LinksDrehen**
    - Drehe dich um 90° nach rechts! — **RechtsDrehen**
    - Setze/Lösche eine Marke! — **MarkeSetzen / MarkeLöschen**
    - Lege einen Ziegel ab! — **Hinlegen**
    - Hebe einen Ziegel auf! — **Aufheben**
  - er seine Umwelt wahrnimmt:
    - Wohin sehe ich? — **IstNorden / IstSüden**
    - Stehe ich vor einer Wand? — **IstWand**
    - Stehe ich vor einem Ziegel? — **IstZiegel**
    - Stehe ich auf einer Marke? — **IstMarke**
    - Ist mein Rucksack voll oder leer? — **IstVoll / IstLeer**

1. Aufgabe

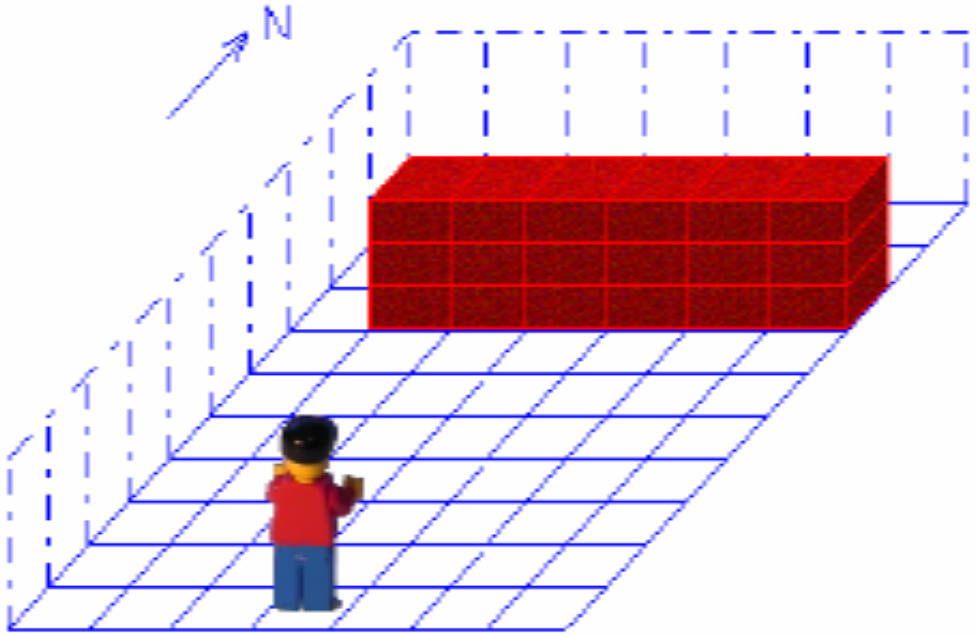
- **Karol steht auf dem Hof vor einer Mauer. Er soll bis zur Mauer laufen und davor stehen bleiben.**

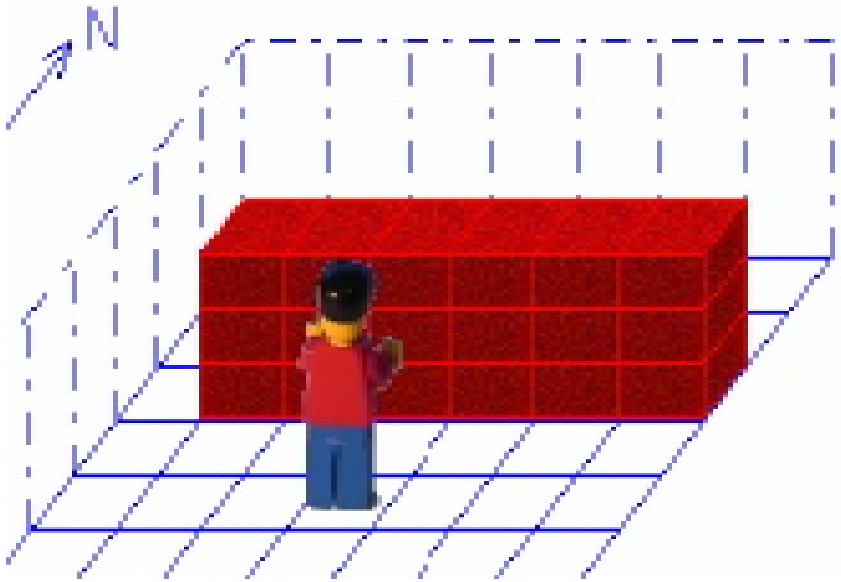


  - **Beschreibe in Worten, was Karol machen soll.**

    - **Karol geht 5 Schritte und hält an**

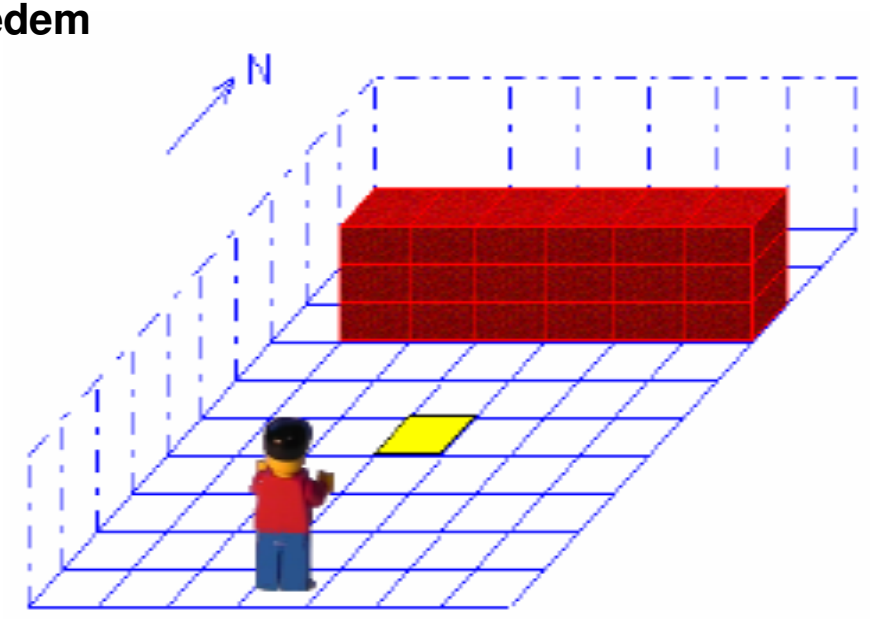
- **Was passiert, wenn Karol NICHT 5 Felder vor der Wand steht?**





solange NichtIstZiegel tue
Schritt
***solange**

- **Karol steht auf dem Hof und läuft gerade bis zur Wand (Aufgabe 1). Wenn er unterwegs eine Marke findet, soll er sie aufheben.**
  - **Welche Funktion muss Karol bei jedem Feld zusätzlich anwenden?**




```
solange NichtIstZiegel tue
        Schritt
        wenn IstMarke dann
                MarkeLöschen
        *wenn
*solange
```

Kontrollstrukturen

wenn *Bedingung* dann *Anweisungen* *wenn	bedingte Anweisung
wenn *Bedingung* dann *Anweisungen* sonst *Anweisungen* *wenn	bedingte Anweisung zweiseitig
wiederhole *Zahl* mal *Anweisungen* *wiederhole	gezählte Wiederholung
solange *Bedingung* tue *Anweisungen* *solange	bedingte Wiederholung mit Anfangsprüfung
wiederhole solange *Bedingung* *Anweisungen* *wiederhole	bedingte Wiederholung mit Anfangsprüfung
wiederhole *Anweisungen* *wiederhole solange *Bedingung*	bedingte Wiederholung mit Endprüfung

Aufgaben

Fachhochschule Jena
University of Applied Sciences Jena

Für folgende Probleme soll ein Algorithmus in Form eines Struktogramms erarbeitet werden:

- **Karol soll seine Welt aufräumen. Deshalb soll er alle Steine, die rumliegen, einsammeln.**
  Hinweis: aufheben(), IstZiegel

- **Karol soll in die linke obere Ecke gehen und nach Süden schauen, egal, wo er sich gerade befindet.**
  Hinweis: IstSüden

- **Karol soll die gesamte Welt mit einer Schicht Steine auslegen**

- **Karol soll eine Burg bauen, die aus einer Mauer aus 4 Schichten und Zinnen besteht.**

- **Weitere Übungen unter:**
  http://www.bics.be.schule.de/inf2/didaktik/minilanguages/aufgaben/uebungsaufgaben_robot_karol.html

Struktogramm

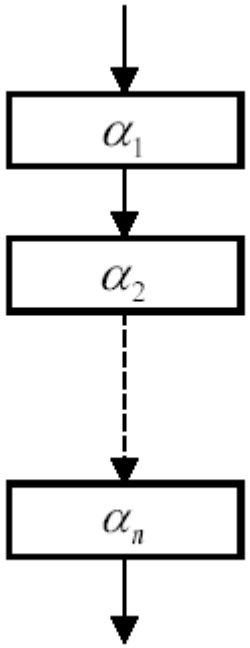
- Grundelement: Strukturblock – dargestellt als Rechteck
- Strukturblock enthält Anweisung(en)
- Abarbeitungsfluß stellt sich durch *waagerechte, gemeinsame* Rechteckseiten dar
- Entstanden mit „strukturierter Programmierung“ (d.h. kein „goto“)

- NachfolgendGegenüberstellung von Pseudocode, Ablaufplan, Struktogramm

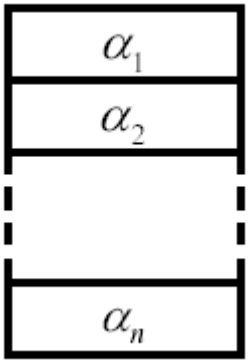
Anweisungssequenz

Fachhochschule Jena
University of Applied Sciences Jena

PAP



Struktogramm

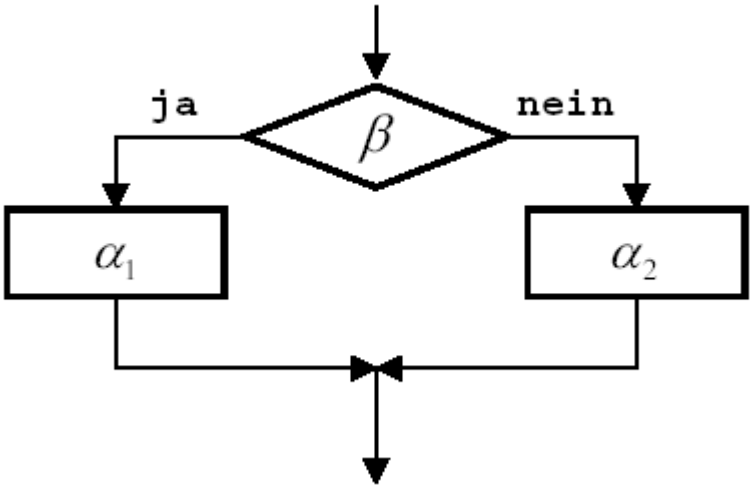


Pseudocode und C:

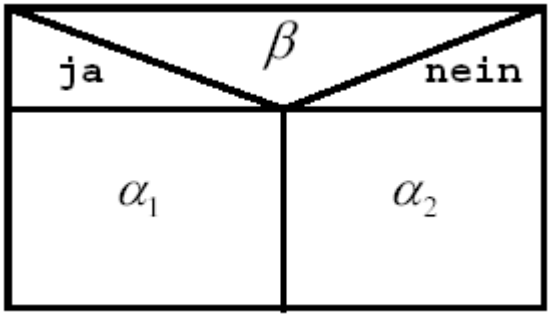
α 1; α 2; α 3; ....; α n

Zeichen ; als „Sequenzierungsoperator“

PAP



Struktogramm

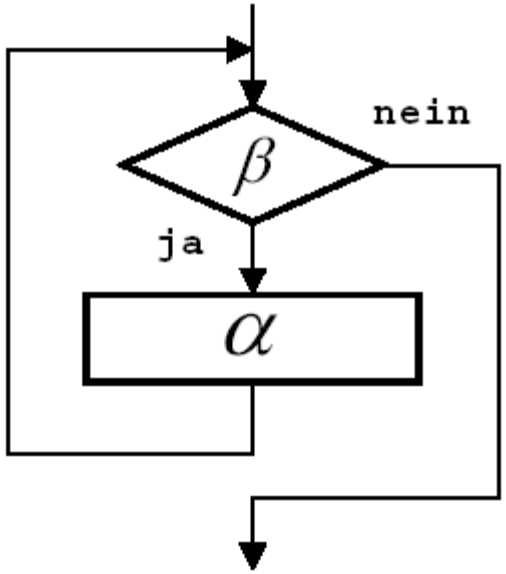


Pseudocode:

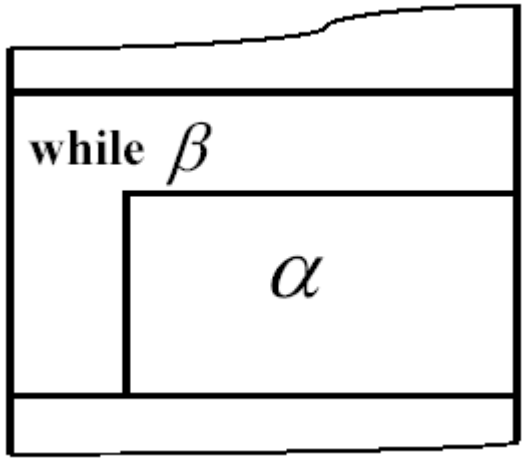
wenn β dann
    α 1;
sonst
    α 2;

Abweisende Wiederholungsanweisung

PAP



Struktogramm

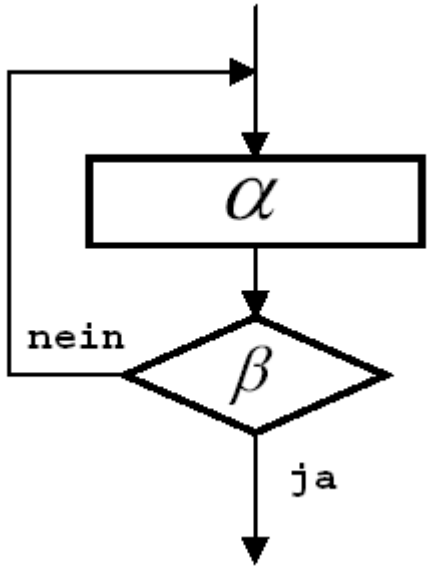


Pseudocode:

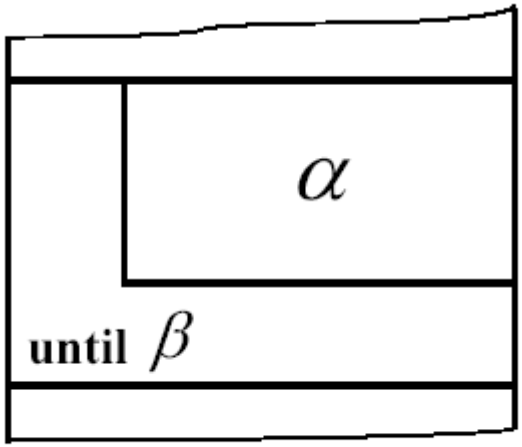
solange β tue

α ;

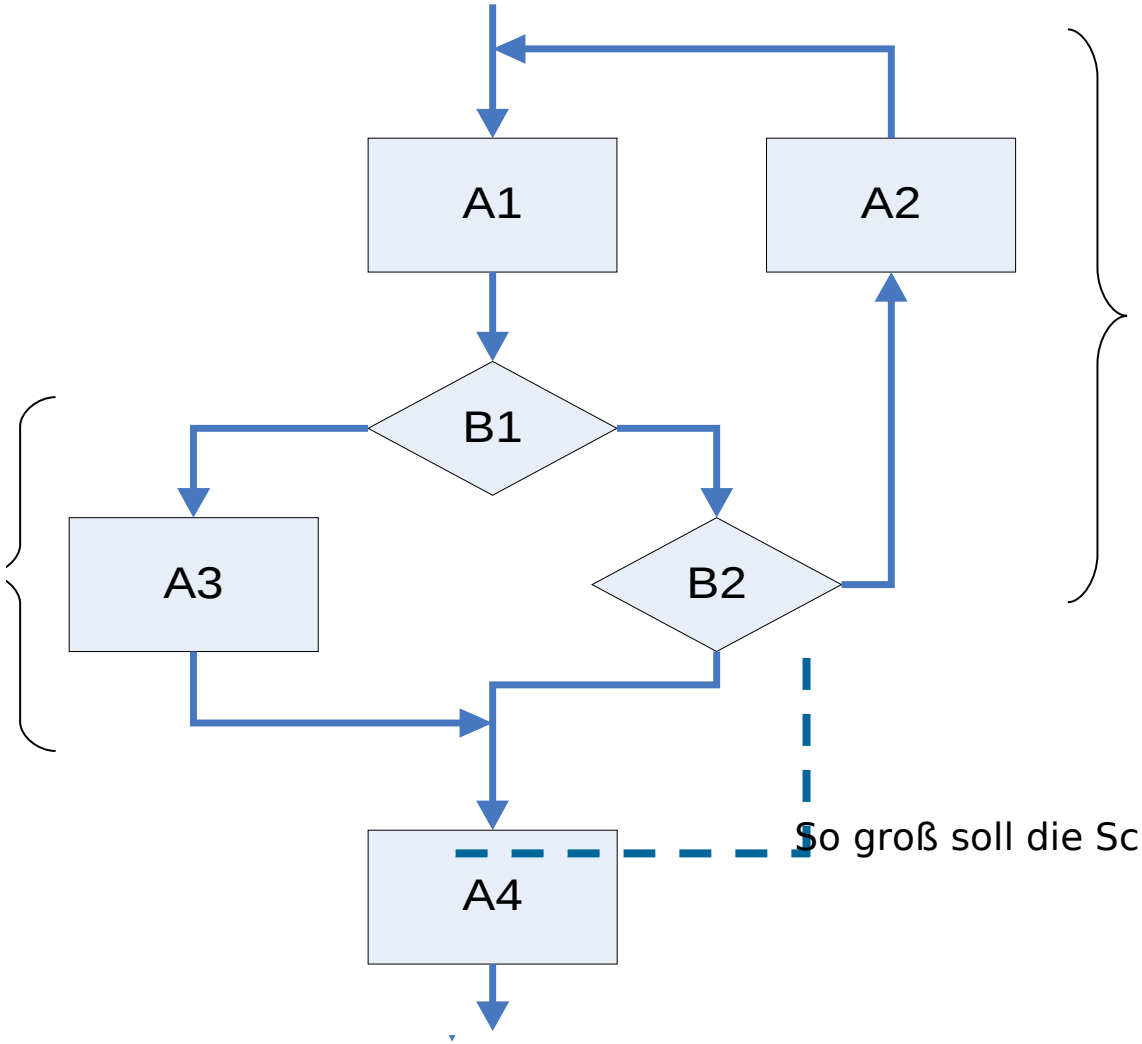
PAP



Struktogramm



Pseudocode:
wiederhole
α
bis β



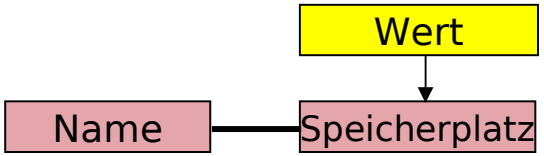
Wirkungsbereich der Schleife

Wirkungsbereich der Alternative

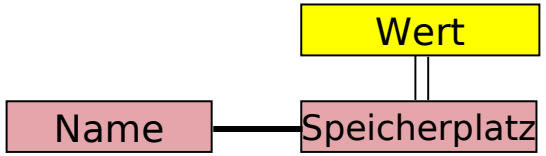
So groß soll die Schleife werden!

Variablen und Konstanten

- Daten (Werte) eines Programms in höherer Programmiersprache werden in Variablen gespeichert.
- Variable ist ein benannter Behälter, d.h. eine Speicherzelle mit Namen für einen Wert
  (anstatt Speicherzelle auch Speicherplatz, oder kurz Zelle;
  anstatt Name auch Bezeichner)
- Unterschied zur Mathematik:
  - Mathe: Bindung Name ↔ Wert
  - Programmierung: Bindung Name ↔ Zelle
    In Zelle können im Verfahrensablauf verschiedene Werte stehen!
- Größe und Form des Behälters durch Typ festgelegt



- Konstanten - ein fester Wert
- Benannte Konstante, z.B. π , e
  - ähnelt der Variablen, aber der Wert ist (meist) unveränderlich



- **Zweck: Berechne aus (arithmetischem, logischem, …) Ausdruck einen Wert und speichere diesen Wert in einer angegebenen Variable.**

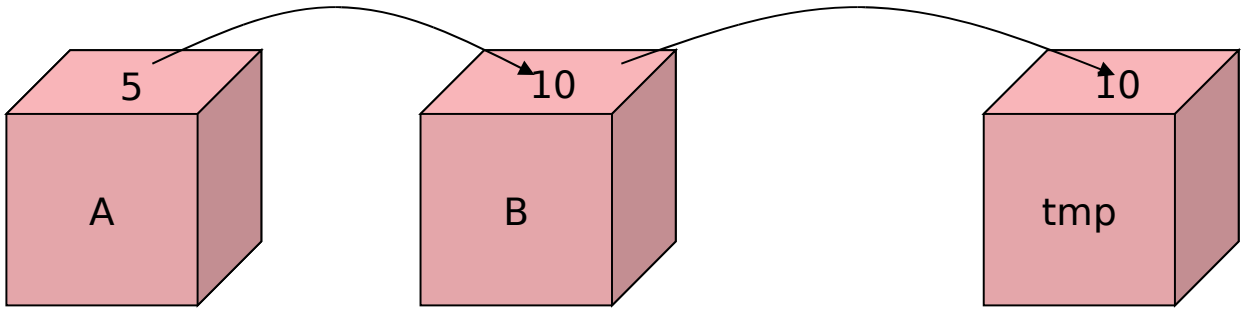
- **Beispiele:**
  - **y ← x + 1 „Der Wert der Variablen y ergibt sich aus dem Wert des Ausdrucks x + 1.“**
  - **x ← x + 1 „Der Wert der Variablen x ergibt sich aus dem Wert des Ausdrucks x + 1 d.h. überspeichert Wert in x.“**

- **Falsche Zuweisungen:**
  - **3 ← y** **links muss Variable stehen! Also keine Konstante**
  - **x + y ← x + 1** **oder kein Ausdruck**

Beispiel: Vertauschen zweier Variablen

- A ← 5
- B ← 10

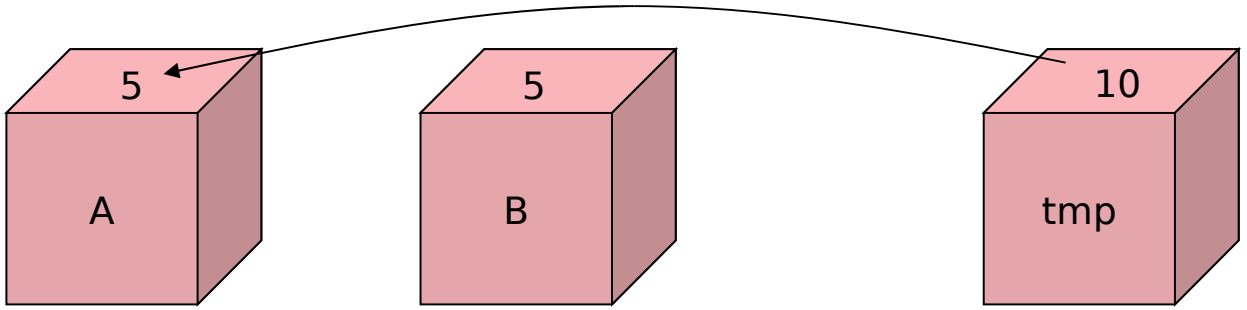
Wie vertausche ich die Inhalte der Variablen, so dass in A der Wert von B (10) und in B der Wert von A (5) steht?



Beispiel: Vertauschen zweier Variablen

- A ← 5
- B ← 10

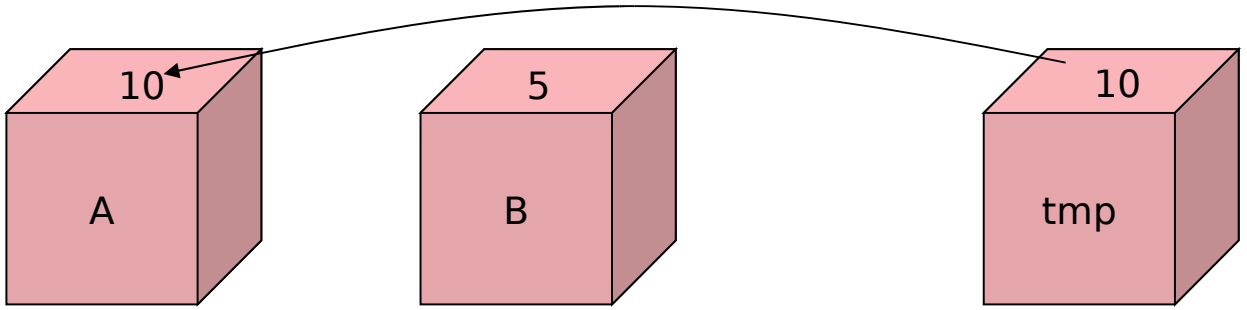
Wie vertausche ich die Inhalte der Variablen, so dass in A der Wert von B (10) und in B der Wert von A (5) steht?



Beispiel: Vertauschen zweier Variablen

- A ← 5
- B ← 10

Wie vertausche ich die Inhalte der Variablen, so dass in A der Wert von B (10) und in B der Wert von A (5) steht?



Zusammenfassung

- **Programmerstellung ist**
  - **Analyse**
  - **Spezifikation**
  - **Definition der Datenstruktur**
  - **Algorithmusdefinition**
  - **Programmierung**

- **Elemente der Programmierung**
  - **Variable: Behälter für veränderliche Daten, der durch einen eindeutigen Namen identifiziert werden kann.**
  - **Konstante: Behälter für konstante Daten.**
  - **Ausdruck: rechte Seite einer Wertzuweisung**
  - **Anweisung: sinnvoll zusammengesetzte Befehlsfolge**
  - **Verzweigung: Gabelung des Programmflusses**
  - **Wiederholung: Schleife im Programmfluss**

Zusammenfassung II

Fachhochschule Jena
University of Applied Sciences Jena

In dieser Vorlesung beschäftigen wir uns mit

- **Datenstrukturen:**
  - **Wie können Informationen als Daten erfasst werden?**
  - **Welche Eigenschaften haben die entstehenden Datenelemente?**
  - **Welche Beziehungen haben die Datenelemente untereinander?**
- **Algorithmen:**
  - **Wie kann das zu lösende Problem auf den Datenstrukturen formalisiert und beschrieben werden?**
- **der Programmiersprache C:**
  - **C als Handwerkszeug eines Ingenieurs**

- **Syntax: formale Regeln für Satzmuster**
  - „Der Elefant aß die Erdnuß.“ (syntaktisch korrekt)
  - „Der Elefant aß Erdnuß die.“ (syntaktisch falsch)
  - Grammatik: Regelwerk für Syntax
    - Regel ist Bildungs- und Analysevorschrift für Sätze oder Teilsätze
    - Bsp: Satz ::= Subjekt Prädikat Objekt .
- **Semantik: Bedeutung**
  - „Der Elefant aß die Erdnuß.“ (semantisch korrekt)
  - „Die Erdnuß aß den Elefanten.“ (semantisch falsch)