Einführung in die Programmierung mit Skriptsprachen

V01: Skriptsprachen, Programmierung, Algorithmen

Dipl.-Inf. BC George

24.10.16

FH Bielefeld / Fachbereich Technik

Agenda:

Zeitplan

Literatur

Skripte und Skriptsprachen

Programme und Programmiersprachen

Compiler und Interpreter

Algorithmen

- Pseudocode
- Struktogramme
- Strukturierte Programmierung

Elemente von Programmiersprachen

Aufgaben und Fragen

Optimistischer Zeitplan

KW	Vorlesung	Praktikum
43 (24.10.)	1: Skriptsprachen, Programmierung, Algorithmen, Bl. 1	-
44 (31.10.)	2: Erste Schritte in Python. Bl. 2	Bl. 1
45 (7.11.)	3: Stack, Objektreferenzen, Datentypen, Bl. 3	Bl. 2
46 (14.11.)	4: Operatoren, Syntax, Semantik, Textdateien, Fehlerbehandlung, Bl. 4	Bl. 3
47 (21.11.)	5: Funktionen, Parameter, Rekursion, Bl. 5	Bl. 4
48 (28.11.)	6: Namensräume, Gültigkeitsbereiche, OOP, Bl. 6	Bl. 5
49 (5.12)	7: GUI mit tkinter, Bl. 7	Bl. 6
50 (12.12.)	8: Einsatzgebiete und Struktur von XML, Bl. 8	Bl. 7
51 (19.12.)	9: XML - Programmierbibliotheken, Bl. 9	Bl. 8
1 (2.1.)	10: Programmierwoche, -	-
2 (9.1.)	10: XML - DTD, Bl. 10	Bl. 9+10
3 (16.1.)	11: XML - Schema, -	*

* = Besprechung der Probeklausur und Spickzettelerstellung

Literatur

Zum Erlernen einer Programmiersprache braucht man

- ein Lehrbuch

Zum Programmieren braucht man

- eine Sprachdefinition / Referenz

Literaturliste

Einführende Literatur

- Theis, Th., Einstieg in Python, Bonn, 2011
- Barry, P., Python von Kopf bis Fuß, Köln, 2011
- Heppert, L., Coding for Fun mit Python, Bonn, 2010
- Vonhoegen, H., Einstieg in XML, Bonn, 2011
- Jones, C. A., Drake, F. L. Jr., Python & XML, Sebastopol, 2002

Referenz

- Summerfield, M., A complete Introduction to the Python Language (Developer‘s Library), Amsterdam, 2009
- Beazley, D. M., Python Essential Reference (Fourth Edition), Amsterdam, 2009
- St. Laurent, S., Fitzgerald, M., XML kurz und gut, Köln, 2006

Online-Dokumentation

- http://docs.python.org/py3k/
- http://www.w3.org/TR/2008/REC-xml-20081126/

Das könnte ja auch ein Computer tun

Aufgabe: Auf einem Rechner für jede/n der 67 Studierenden ein Unterverzeichnis anlegen und die Dateien <datei1> und <datei2> hineinkopieren.

```
mkdir <name1>
cp <date1>, <datei2> <name1>

mkdir <name2>
cp <date1>, <datei2> <name2>

mkdir <name3>
cp <date1>, <datei2> <name3>

mkdir name4>
cp <date1>, <datei2> <name4>
...
```

Wie können wir uns das Leben leichter machen?

Skripte

Skripte

Skripte sind eine Abfolge von Befehlen, die nacheinander ausgeführt werden.

- stehen oft in Dateien
- steuern Betriebssysteme oder Programme
- rufen Programme auf

Skriptsprachen

Definition

Eine **Skriptsprache** ist Programmiersprache, mit der man Skripte schreiben oder ausführen kann.

Die Programme, die man in einer Skriptsprache formuliert, werden oft auch Skripte genannt.

Skriptsprachen werden auch für eher kleine, überschaubare Programme eingesetzt, nicht nur für Skripte.

Skripte bei der Systemverwaltung

- regelbasiertes Erzeugen von Befehlstexten
- automatisierte Erzeugung von Skripten
- bedingte und berechnete Inhalte der Skripte
- Ad-hoc Automatisierung repetitiver Aufgaben
- Automatisieren von Aufgaben in problematischen Zeitfenstern (Wer schubst einen Job freiwillig nachts um halb drei am Sonntag an?)
- Regelbasiertes Ausführen unterschiedlicher Aufgaben

Automatisiertes Erzeugen von Skripten...

...durch Scripting

Anforderungen:

- einfache Umgebung
- einfache Syntax
- schnelle Ergebnisse
- sehen, was man tut
- kein kompliziertes Kompilieren
- keine Lizenzgebühren
- viele fertige Lösungen
- gute Dokumentation
- gute Community

Programme

Definition

Ein **(Computer-)Programm** ist eine Folge von mittels eines Computers ausführbaren Anweisungen zur Erledigung einer bestimmten Aufgabe.

Definition

Programmierung (=Implementierung) ist das Erstellen eines (lauffähigen) Programms.

Programme aus Nullen und Einsen?

Ein Computer kann nur **Maschinencode** direkt ausführen:

```
 00101011 11101010 ... (siehe Technische Informatik)
```

... nicht sehr gut lesbar für Menschen...

Programme aus Nullen und Einsen?

Ein Computer kann nur **Maschinencode** direkt ausführen:

```
 00101011 11101010 ... (siehe Technische Informatik)
```

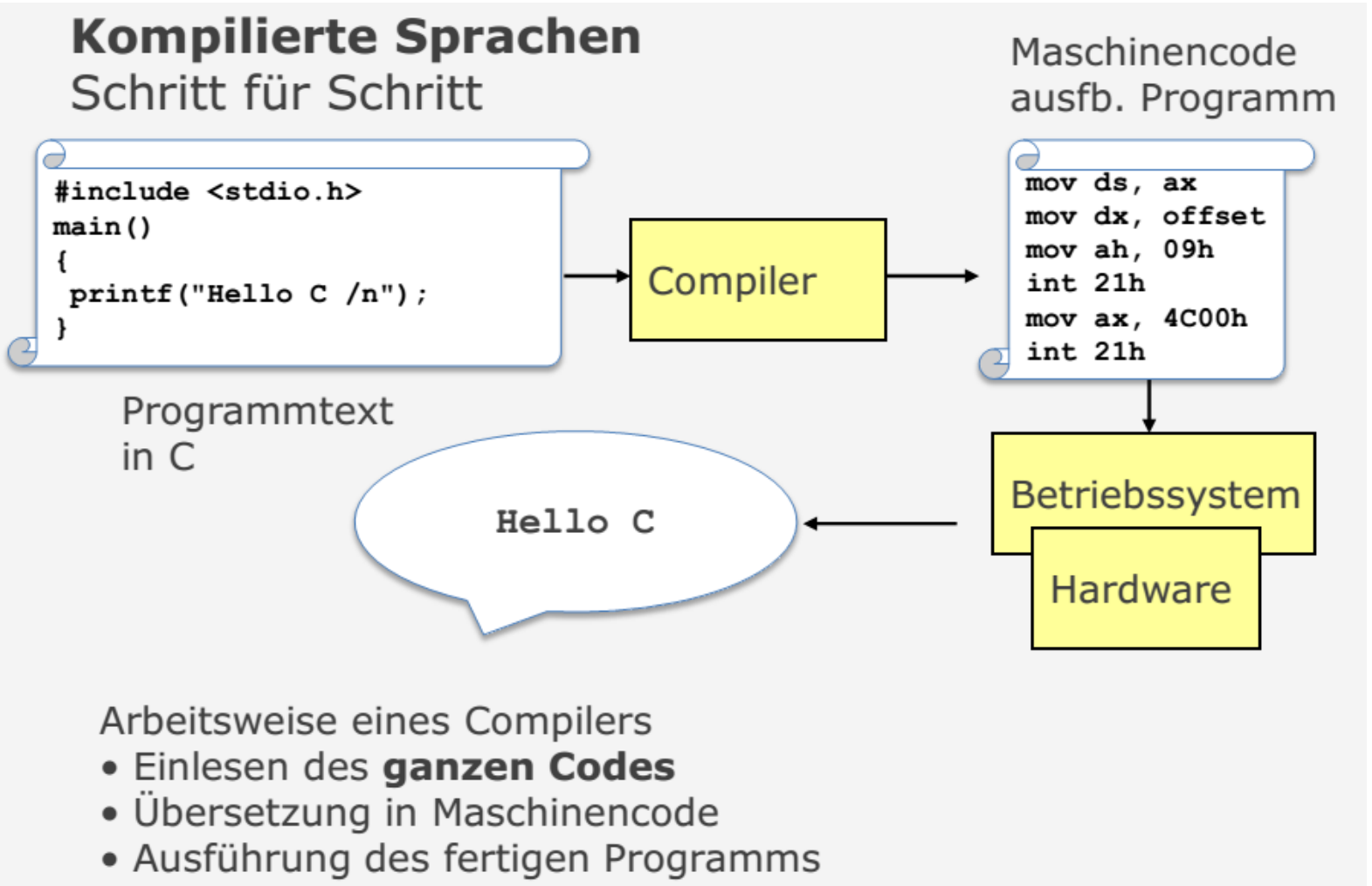
... nicht sehr gut lesbar für Menschen...

Programmierer schreiben die Programme in **Hochsprachen**, die es für verschiedene Zwecke gibt.

Hochsprachen

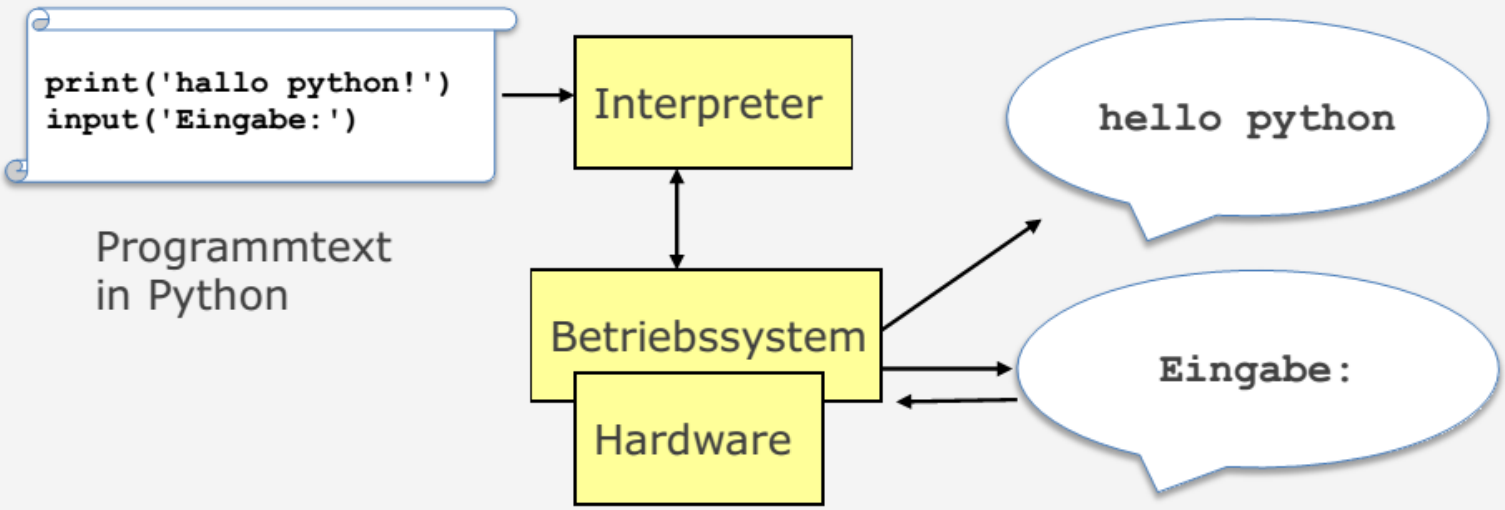
- **Kompilierte Programme:** Ein **Compiler** übersetzt ein Programm aus einer bestimmten Hochsprache in eine bestimmte Maschinensprache. Das Programm in Maschinensprache wird vom Computer ausgeführt.
  oder
- **Interpretiere Programme:** Ein **Interpreter** liest ein Programm in einer bestimmten Hochsprache zeilenweise und führt jede Zeile direkt aus.
  oder
- Ein Compiler übersetzt ein Hochsprachenprogramm in maschinennahen, aber maschinenunabhängigen **Bytecode**, der dann von einer virtuellen Maschine auf der jeweiligen Plattform interpretiert wird.

Compiler


Kompilierte Sprachen
Schritt für Schritt
#include <stdio.h>
main()
{
printf("Hello C /n");
}
Compiler
Maschinencode ausfb. Programm
mov ds, ax
mov dx, offset
mov ah, 09h
int 21h
mov ax, 4C00h
int 21h
Programmtext in C
Hello C
Betriebssystem
Hardware
Arbeitsweise eines Compilers
• Einlesen des ganzen Codes
• Übersetzung in Maschinencode
• Ausführung des fertigen Programms

Interpreter

Interpretierte Sprachen
Schritt für Schritt



Jedes Betriebssystem benötigt einen eigenen Interpreter

einfaches Erlernen durch Experimentieren Zeile für Zeile möglich

Compiler und Interpreter

Interpretierte Sprachen	**Kompilierte Sprachen**
Direkte Ausführung Zeile für Zeile	Übersetzung des Programmcodes in ausführbare Dateien
Kein Deklarationszwang	Variablen und andere Elemente müssen deklariert werden
Dynamische Typisierung	Mehr oder weniger strikte Typisierung
Relativ plattformunabhängig	Großer Sprachumfang
Speicherverwaltung u. –bereinigung automatisch	Komplexe Entwicklungsumgebungen
Oft frei verfügbar	

Skriptsprachen sind in der Regel interpretierte Sprachen.

Wie stellen Sie sich die Arbeit mit einem Interpreter, bzw. Compiler, vor?

Wie stellen Sie sich die Arbeit mit einem Interpreter, bzw. Compiler, vor?
Welches sind die Vor- und Nachteile von Interpretern, bzw. Compilern, bzgl. der Geschwindigkeit?

Wie stellen Sie sich die Arbeit mit einem Interpreter, bzw. Compiler, vor?
Welches sind die Vor- und Nachteile von Interpretern, bzw. Compilern, bzgl. der Geschwindigkeit?
Wie aufwendig ist die Portierung einer interpretierten, bzw. kompilierten Sprache auf ein neues Betriebssystem?

Pause

Algorithmus

Und wenn ich meinen Programmablauf nicht in einer bestimmten Hochsprache formulieren will oder kann? (Warum?)

Algorithmus

Und wenn ich meinen Programmablauf nicht in einer bestimmten Hochsprache formulieren will oder kann? (Warum?)

Definition

Ein **Algorithmus** ist eine aus endlich vielen Schritten bestehende eindeutige Handlungsvorschrift zur Lösung eines Problems oder einer Klasse von Problemen. (Wikipedia)

Algorithmus

Und wenn ich meinen Programmablauf nicht in einer bestimmten Hochsprache formulieren will oder kann? (Warum?)

Definition

Ein **Algorithmus** ist eine aus endlich vielen Schritten bestehende eindeutige Handlungsvorschrift zur Lösung eines Problems oder einer Klasse von Problemen. (Wikipedia)

Jetzt sind Sie dran!

Suchen Sie Beispiele für Algorithmen aus dem täglichen Leben.

Wie werden Algorithmen aufgeschrieben?

Algorithmen werden **von Menschen** entwickelt, notiert, gelesen, bearbeitet, weitergegeben, ... und in Programme umgewandelt.

- Sie müssen möglichst intuitiv verständlich sein.
- Sie sollten möglichst dieselben Strukturelemente enthalten wie Programmcode.

Notation

- **Pseudocode:** oft angelehnt an die benutzte Programmiersprache
- **Struktogramme (Nassi-Shneidermann-Diagramme):** Sinnbilder nach DIN 66261

Muffins backen in Pseudocode

```
schüssel = 150g Zucker + 250g Butter + 4 EL
    Milch
rühre 5 min
schüssel = schüssel + 1 Ei
rühre 2 min
schüssel = schüssel + 1 Ei
rühre 2 min
schüssel = schüssel + 1 Ei
rühre 2 min
schüssel = schüssel + 1 Ei
rühre 5 min
schüssel = schüssel + 250g Mehl + 1 TL
    Backpulver
rühre 5 min
```

◂ Alg2 ▸ Elemente

Was sagen Sie dazu?

- Ist das ein Algorithmus?

  ◂ Alg1

Was sagen Sie dazu?

- Ist das ein Algorithmus?

  ◂ Alg1

- Kann man damit eine Klasse von Problemen lösen?

  ◂ Alg1

Was sagen Sie dazu?

- Ist das ein Algorithmus?

  ◂ Alg1

- Kann man damit eine Klasse von Problemen lösen?

  ◂ Alg1

- Was können wir ändern, damit man unterschiedliche Mengen von Muffins backen kann?

Muffins - Programm 2

Zutaten:

zucker = 200g Zucker
butter = 150g Butter
anzahleier = 4
milch = 4TL Milch
mehl = 250g Mehl
backpulver = 1 Teelöffel

Algorithmus:

◂ Alg1

Muffins - Programm 2

Zutaten:

zucker = 200g Zucker
butter = 150g Butter
anzahleier = 4
milch = 4TL Milch
mehl = 250g Mehl
backpulver = 1 Teelöffel

Algorithmus:

schüssel = zucker + butter + milch

◂ Alg1

Muffins - Programm 2

Zutaten:

zucker = 200g Zucker
butter = 150g Butter
anzahleier = 4
milch = 4TL Milch
mehl = 250g Mehl
backpulver = 1 Teelöffel

Algorithmus:

schüssel = zucker + butter + milch

◂ Alg1

repeat
    rühre(1 min)

▸ Maßnahmen

until teig einheitlich

Muffins - Programm 2

Zutaten:

zucker = 200g Zucker
butter = 150g Butter
anzahleier = 4
milch = 4TL Milch
mehl = 250g Mehl
backpulver = 1 Teelöffel

Algorithmus:

```
schüssel = zucker + butter + milch

repeat
    rühre(1 min)
until teig einheitlich

for i = 1 to anzahleier - 1
    schüssel = schüssel + i-tes ei
    rühre (2 min)
```

◂ Alg1

▸ Maßnahmen

▸ Elemente

▸ nur3

Muffins - Programm 2

Zutaten:

zucker = 200g Zucker
butter = 150g Butter
anzahleier = 4
milch = 4TL Milch
mehl = 250g Mehl
backpulver = 1 Teelöffel

Algorithmus:

```
schüssel = zucker + butter + milch
repeat
    rühre(1 min)
until teig einheitlich
for i = 1 to anzahleier - 1
    schüssel = schüssel + i-tes ei
    rühre (2 min)
if teig zu fest
    schüssel = schüssel + ei
    rühre (2 min)
else
    rühre (1 min)
```

◂ Alg1

▸ Maßnahmen

▸ Elemente

▸ nur3

Muffins - Programm 2

Zutaten:

zucker = 200g Zucker
butter = 150g Butter
anzahleier = 4
milch = 4TL Milch
mehl = 250g Mehl
backpulver = 1 Teelöffel

Algorithmus:

```
schüssel = zucker + butter + milch
repeat
    rühre(1 min)
until teig einheitlich
for i = 1 to anzahleier - 1
    schüssel = schüssel + i-tes ei
    rühre (2 min)
if teig zu fest
    schüssel = schüssel + ei
    rühre (2 min)
else
    rühre (1 min)
schüssel = schüssel + backpulver
```

◂ Alg1

▸ Maßnahmen

▸ Elemente

▸ nur3

Muffins - Programm 2

Zutaten:

zucker = 200g Zucker
butter = 150g Butter
anzahleier = 4
milch = 4TL Milch
mehl = 250g Mehl
backpulver = 1 Teelöffel

Algorithmus:

```
schüssel = zucker + butter + milch
repeat
    rühre(1 min)
until teig einheitlich
for i = 1 to anzahleier - 1
    schüssel = schüssel + i-tes ei
    rühre (2 min)
if teig zu fest
    schüssel = schüssel + ei
    rühre (2 min)
else
    rühre (1 min)
schüssel = schüssel + backpulver
while teig uneinheitlich
    rühre (1 min)
```

◂ Alg1

▸ Maßnahmen

▸ Elemente

▸ nur3

Was haben wir gemacht?

- die Mengenangaben vom eigentlichen Algorithmus getrennt
- *rühre* mit einem Parameter versehen
- *while*, *repeat*, *for*, *if*, *else* benutzt

◂ Alg2

Was haben wir gemacht?

- die Mengenangaben vom eigentlichen Algorithmus getrennt
- *rühre* mit einem Parameter versehen
- *while*, *repeat*, *for*, *if*, *else* benutzt

◂ Alg2

Was haben wir damit erreicht?

Was haben wir gemacht?

- die Mengenangaben vom eigentlichen Algorithmus getrennt
- *rühre* mit einem Parameter versehen
- *while*, *repeat*, *for*, *if*, *else* benutzt

◂ Alg2

Was haben wir damit erreicht?
Welche Rolle spielt die Einrückung?

Strukturelemente von Algorithmen

Definition

Kontrollstrukturen sind Mechanismen in einem Algorithmus oder Programm, die festlegen, in welcher Reihenfolge die einzelnen Befehle des Algorithmus oder Programms ausgeführt werden.

Dieselben Kontrollstrukturen können in allen Algorithmen vorkommen.

Strukturelemente von Algorithmen

Definition

Kontrollstrukturen sind Mechanismen in einem Algorithmus oder Programm, die festlegen, in welcher Reihenfolge die einzelnen Befehle des Algorithmus oder Programms ausgeführt werden.

Dieselben Kontrollstrukturen können in allen Algorithmen vorkommen.

Welche Kontrollstrukturen finden Sie in den Muffins-Rezepten?

◂ Alg1 ◂ Alg2

Kontrollstrukturen in Pseudocode

- **Fallunterscheidung**

```
if <bedingung> then
  <anweisungsblock>
oder
if <bedingung> then
  <anweisungsblock>
else
  <anweisungsblock>
```

- **zählergesteuerte Schleife**

```
for <variable> = <wert1> to <wert2> do
  <anweisungsblock>
```

- **while – Schleife**

```
while <bedingung> do
  <anweisungsblock>
```

- **repeat until – Schleife**

```
repeat
  <anweisungsblock>
until <bedingung>
```

... und was gibt es noch?

Befehle, die nicht die Reihenfolge der Abarbeitung bestimmen, sondern die eigentlichen Aktionen durchführen.

Feststellung

Algorithmen bestehen aus **Kontrollstrukturen** und **Elementaroperationen**.

Die Elementaroperationen, die in einem Algorithmus vorkommen, werden bestimmt durch:

- dem Zweck des Algorithmus
- dem Ausführenden des Algorithmus
- allgemein: dem zugrundeliegenden Rechenmodell

Die Elementaroperationen eines Algorithmus müssen z. B. für den Ausführenden eindeutig sein. Elementaroperationen sind nicht universell wie Kontrollstrukturen.

Elementaroperationen in Computerprogrammen

Wichtige Elementaroperationen bei Computern sind **Zuweisungen**. Sie schreiben Werte in Speicherzellen. Speicherzellen haben in der Regel Namen, z. B. a.

Beispiel:

```
a = 10
b = a - 1
c = c * 2
```

Zuweisungen erkennt man am **Zuweisungszeichen (=)**.

Auf der echten Seite des = steht jeweils ein **Ausdruck**.

Ausdrücke beinhalten **Operatoren**, z. B. mathematische Operatoren wie `+`, `-`, `*`, `/`, ...

Weitere Elemente von Pseudocode

- Erste Zeile: Name des Algorithmus dann evtl. Parameter
- Kommentare:
  einzeilig mit: `# <kommentar>`
  mehrzeilig mit: `/* <kommentar> <kommentar> */`
- Aufrufe von anderen Pseudocode-Modulen mit Parametern und Ergebniswert:
  `b = berechneKreisumfang(r)`

- Zuweisungsoperator / Vergleichsoperator
  `a = 5`
  `if c == 3 then ...`

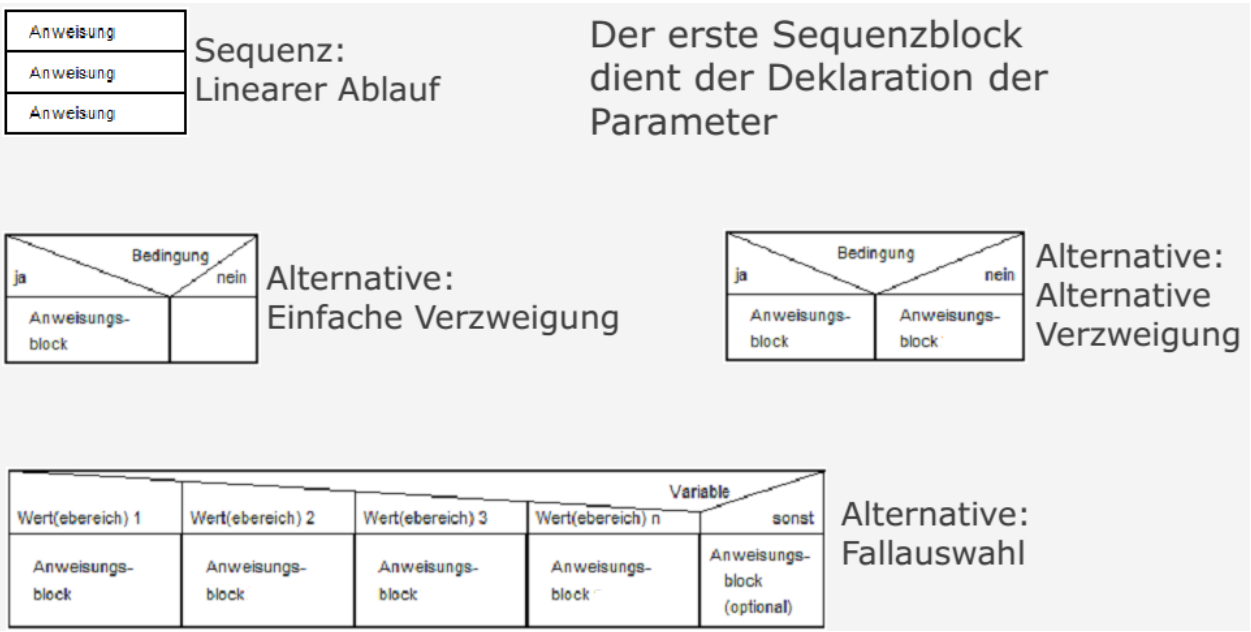
Jetzt sind Sie dran ...

Überlegen Sie sich in kleinen Gruppen einen Algorithmus!

Nassi-Shneidermann-Diagramme (Struktogramme)

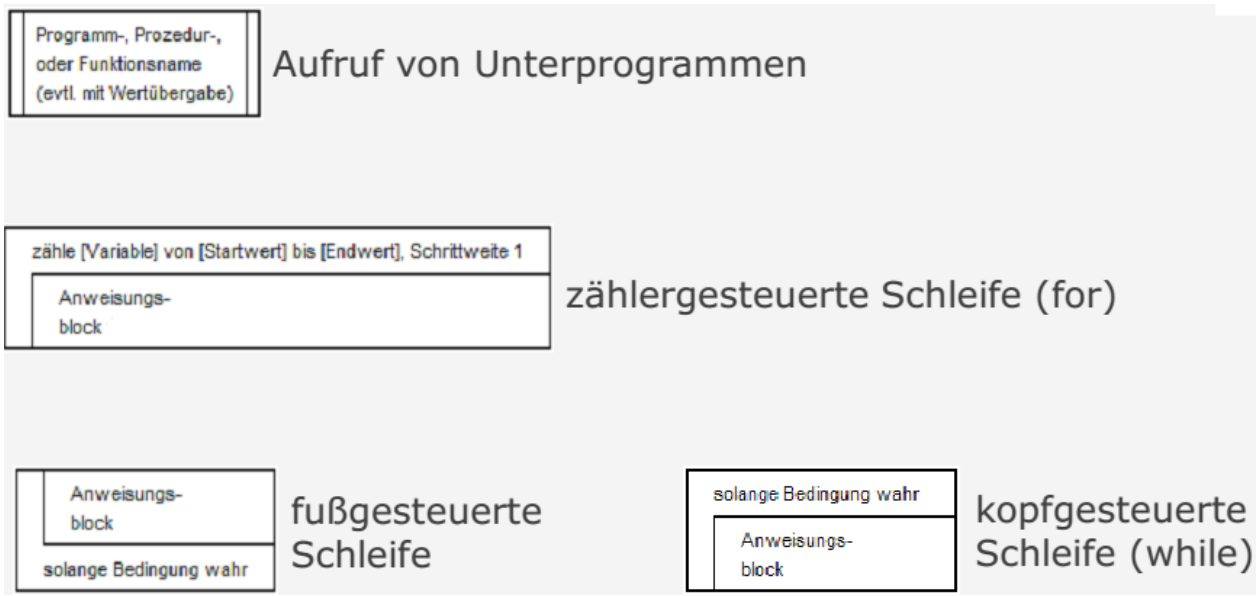
Ein Algorithmus besteht aus einem äußeren Anweisungsblock, der weitere Anweisungsblöcke umfassen kann. Anweisungsblöcke dürfen nur einander folgen (Sequenz), oder ein Block muss komplett in einem anderen enthalten sein.

Anweisungsblocktypen:



Weitere Struktogramme

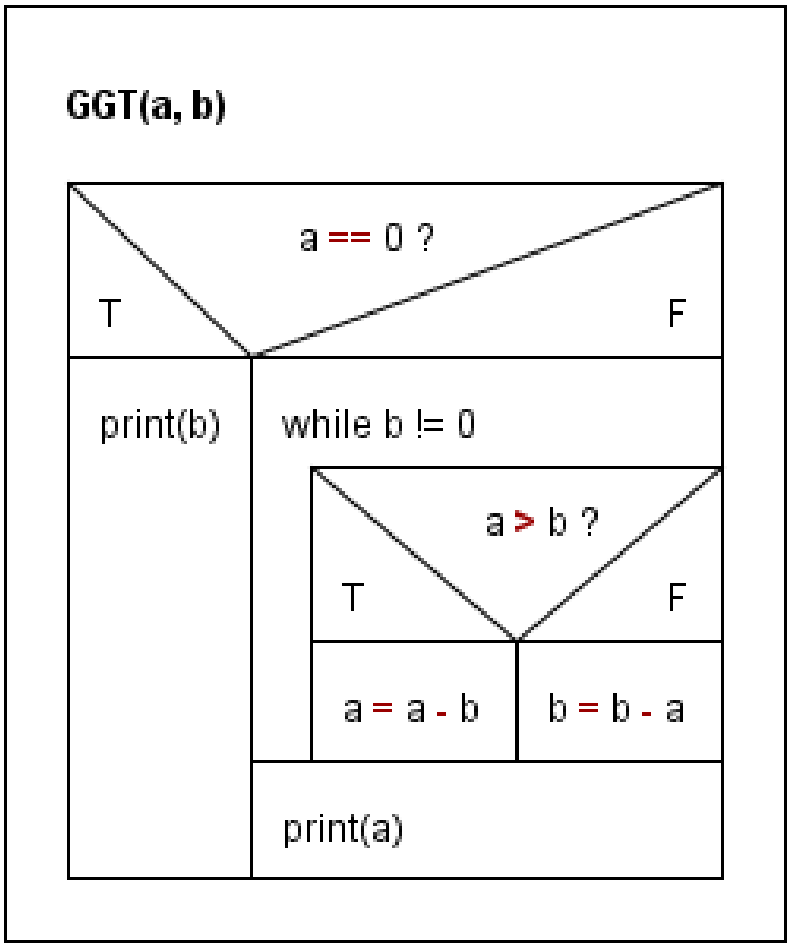
Es gibt folgende weitere Anweisungsblocktypen:



ACHTUNG! GANZ WICHTIG: Die fußgesteuerte Schleife ist anders als bei `repeat ... until`.

Beispiel-Struktogramm

Ein Algorithmus zur Berechnung des GGT (größter gemeinsamer Teiler):



Strukturierte Programmierung

Jeder Algorithmus / jedes Programm lässt sich durch folgende Kontrollstrukturen realisieren:

- **Sequenz:** Ausführung mehrerer Anweisungen hintereinander
- **Auswahl:** Durchführung von Anweisungen in Abhängigkeit bestimmter Bedingungen
- **Wiederholung:** Wiederholte Ausführung einer oder mehrerer Anweisungen in Abhängigkeit von einer Bedingung

Nicht nötig, aber hilfreich:

- **Aufruf anderer Algorithmen:** Anwendung eines anderen Algorithmus (Name und dessen Parameter)

Was gehört zu einer Programmiersprache?

- Schlüsselwörter
- Bezeichner
- Datentypen
- Operatoren
- Kontrollfluss
- Funktionen / Prozeduren
- Input/Output
- GUI (Graphical User Interface)
- Fehlerbehandlung
- Module
- Objektreferenzen
- OOP-Konzept

Aufgabenblatt 1

1. **Karteikarten** **5 Pkt.**
   Erstellen Sie Karteikarten zur Vorlesung. Bringen Sie sie zum nächsten Praktikum mit und begründen Sie Ihre Auswahl.
2. **Struktogramm** **5 Pkt.**
   Erstellen Sie ein Struktogramm für den zweiten Algorithmus zum Backen von Muffins aus der Vorlesung.
3. **Pseudocode** **5 Pkt.**
   Formen Sie den folgenden Algorithmus so um, dass der *if*-Befehl keinen *else*-Teil enthält, der Algorithmus aber dieselbe Funktionalität hat:

```
if teig zu fest
    schüssel = schüssel + ei
    rühre (1 min)
else
    rühre (1 min)
while teig uneinheitlich
    rühre (1 min)
```

Diese Fragen sollten Sie beantworten können:

- Was sind Skriptsprachen?
- Welche Anforderungen werden an Skriptsprachen gestellt?
- Was ist ein Computerprogramm?
- Was ist Maschinencode?
- Welche Konzepte der Übersetzung von Hochsprachprogrammen kennen Sie?
- Was ist ein Algorithmus?
- Wie werden Algorithmen notiert?
- Welche Strukturelemente von Algorithmen und Programmen gibt es?