Designing Education
Connecting People

Das erwartet Sie:

- Daten und Informationen unterscheiden
- Prozess der Softwareentwicklung

Software zur Verwaltung von Daten anpassen

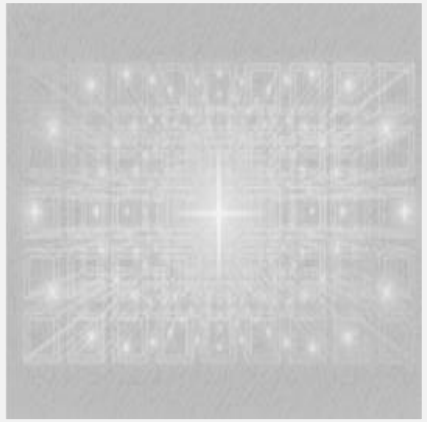
Lernfeld 5

Die Themen und Lernziele

Das Umfeld der Softwareentwicklung analysieren

Lernziel

Aufgaben und Kompetenzen in der SE kennenlernen



Grundlagen zur Verwaltung von Daten

Lernziel

Information versus Daten

Den Prozess der Software-entwicklung analysieren

Lernziel

Prozessphasen sowie Vorgehensmodelle kennenlernen

Den Prozess der Anforderungs-spezifikation beschreiben

Lernziel

Anforderungen an die zukünftige Software spezifizieren können

Einfache Anwendungen in Python schreiben

Lernziel

Programmiersprachen und –werkzeuge unterscheiden lernen

Die Themen und Lernziele

Auf Dateien in Anwendungen zugreifen

Lernziel

Daten speichern und einlesen lernen

Verwaltung der Daten mithilfe von Datenbanken

Lernziel

Grundlagen von relationalen Datenbanken

Software testen und dokumentieren

Lernziel

Qualitätsbewusstsein entwickeln

Prozess der Softwareentwicklung evaluieren

Lernziel

Reflexion

Den Prozess der Anforderungs-spezifikation beschreiben

Lernziel

Anforderungen an die zukünftige Software spezifizieren können

Der heutige Tag

Anforderungen spezifizieren

Lasten – und Pflichten-heft

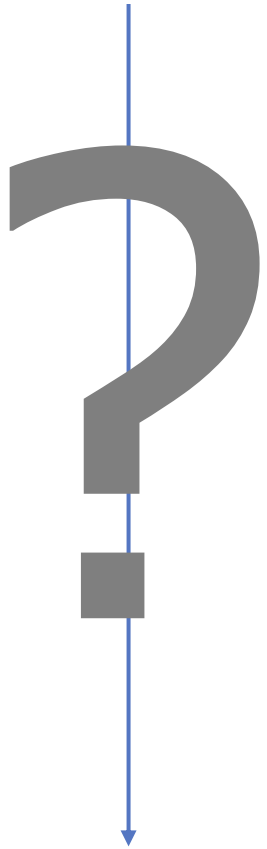
Beschreibung des Entwurfs-prozesses

Modellierung-sprachen

5.4.1. Anforderungen an eine Software spezifizieren

Anforderungsspezifikation (standardisiert vom IEEE)

- Ermitteln
- Analysieren
- Spezifizieren
- Validieren

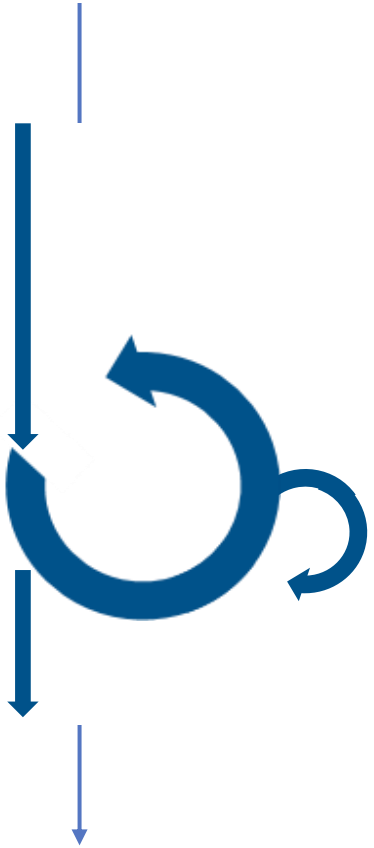


oder

5.4.1. Anforderungen an eine Software spezifizieren

Anforderungsspezifikation (standardisiert vom IEEE)

- Ermitteln
- Analysieren
- Spezifizieren
- Validieren



oder

5.4.1. Anforderungen an eine Software spezifizieren

IEEE [„i-triple e“] 29148-2018:

Ein weltweiter Berufsverband von Ingenieuren aus den Bereichen Elektrotechnik und Informationstechnik.

Er ist Veranstalter von Fachtagungen, Herausgeber div. Fachzeitschriften und bildet Gremien für die Standardisierung von Techniken, Hardware und Software

IEEE

Institute of Electrical and Electronic Engineers

Förderung technologischer Innovationen zum Nutzen der Menschheit.
Sitz: New York, Gegründet Januar 1963

5.4.1. Anforderungsspezifikation

5.4.1. Anforderungen an eine Software spezifizieren

Software

Requirements

Specification (SRS)

- Funktionale
  Was soll das System leisten
- Nicht funktionale
  Wie das System seine Leistung bringen soll

5.4.1. Anforderungen an eine Software spezifizieren

Funktional

- Beschreibung der einzelnen Funktionen und Funktionskomplexe
- Beschreibung der Eingabedaten
- Erforderliche Verarbeitungsschritte
- Erwartete Ausgabe

Nicht funktional

- Qualitätsanforderungen (manche subjektiv)
  - Benutzbarkeit
  - Zuverlässigkeit
  - Effizienz
  - Änderbarkeit
  - Übertragbarkeit
- Randbedingungen (kaum beeinflussbar)
  - Technologisch
  - Organisatorisch
  - Normativ

Kompetenzcheck

Welche Aussagen sind richtig?

a) Der Prozess der Anforderungsspezifikation erfolgt in streng abgegrenzten Schritten.

b) Die Anforderungen müssen eindeutig formuliert sein.

c) Es werden funktionale und nicht funktionale Anforderungen unterschieden.

d) Qualitätsanforderungen sind nicht besonders wichtig, können aber der Vollständigkeit halber erwähnt werden.

e) Die Fehlertoleranz gehört zu den nicht funktionalen Anforderungen.

f) Anforderungen an die Effizienz gehören zu den funktionalen Anforderungen.

g) Kulturelle Aspekte spielen bei der Anforderungsanalyse keine Rolle.

Auftraggeber / Auftragnehmer - Wer ist was?

Ein Fahrgast, setzt sich in ein Taxi und gibt dem Fahrer die Anweisung: "Ich habe es eilig, fahren Sie so schnell, Sie können."

Auf die Frage des Taxifahrers, wohin er denn wolle, antwortet er dann: "Das weiß ich nicht - Sie sind doch der Fahrer!"

5.4.2 Lasten- und Pflichtenheft unterscheiden

Das Lastenheft

Beschreibt die Gesamtheit der Forderungen an die Lieferungen und Leistungen eines Auftragnehmers innerhalb eines (Projekt-)Auftrags" (DIN 69901-5)

- Der Auftraggeber formuliert das Lastenheft
- Dient als Grundlage zur Einholung von Angeboten (*Ausschreibung, Angebotsanfragen*…)

Inhalte des Lastenhefts:

- Die Spezifikation des zu erbringenden Werks
- Die Anforderungen an das Produkt bei seiner späteren Verwendung
- Rahmenbedingungen für Produkt und Leistungserbringung
- Vertragliche Konditionen
- Anforderungen an den Auftragnehmer
- Anforderungen an das Projektmanagement

5.4.2 Lasten- und Pflichtenheft unterscheiden

Das Pflichtenheft

Im Pflichtenheft sind nach DIN 69901-5 die vom Auftragnehmer erarbeiteten Realisierungsvorgaben niedergeschrieben.

- Beschreiben die Umsetzung „des vom Auftraggeber vorgegebenen Lastenheftes“
- Es stellt (*oft in Kombination mit einem Angebot*) die vertragliche Grundlage der zu erfüllenden Leistungen dar

Inhalte des Pflichtenhefts:

Lastenheft zzgl.

- Beschreibung der Lösung
- Durchführungspläne (Projektablauf, Zeit- und Kostenpläne etc.)
- Test- und Prüffunktionen
- Übergabe- und Abnahmebedingungen

5.4.2 Lasten- und Pflichtenheft unterscheiden

Lastenheft versus Pflichtenheft

Anforderungsbeschreibung	Lastenheft	Pflichtenheft
Ersteller	**Auftraggeber**	**Auftragnehmer**
Definition DIN 69905 bzw. DIN 69901-5 VDI-Richtlinien	Gesamtheit der Forderungen an Lieferungen und Leistungen eines Auftragnehmers	Vom Auftragnehmer erarbeitete Realisierungsvorhaben auf Basis des Lastenhefts
Fragestellung	*Was?* und *Wofür?*	*Wie?* und *Womit?*
Detaillierungsgrad	Ergebnisorientiert, allgemein verständlich	Genau spezifiziert, verständlich
Alternative Bezeichnungen	Anforderungsspezifikation; Anforderungskatalog Kundenspezifikation oder requirements specification, Anwenderspezifikation, Fachkonzept, Ausstattungsskizzen	Fachliche Spezifikation, fachliches Feinkonzept, Sollkonzept, funktionelle Spezifikation, Feature specification

5.4.2 Lasten- und Pflichtenheft unterscheiden

Beispiel: Projektantrag Lastenheft / Pflichtenheft

Lastenheft

1. Ausgangssituation
2. Zielsetzungen
3. Produkteinsatz bzw. betroffene Arbeitsplätze und Schnittstellen
4. Funktionale Anforderungen
5. Nicht funktionale Anforderungen
6. Lieferumfang
7. Phasenplanung, Meilensteine
8. Offene noch zu klärende Punkte
9. Abnahmekriterien und Qualitätsanforderungen
10. Zuständigkeiten für dieses Projekt

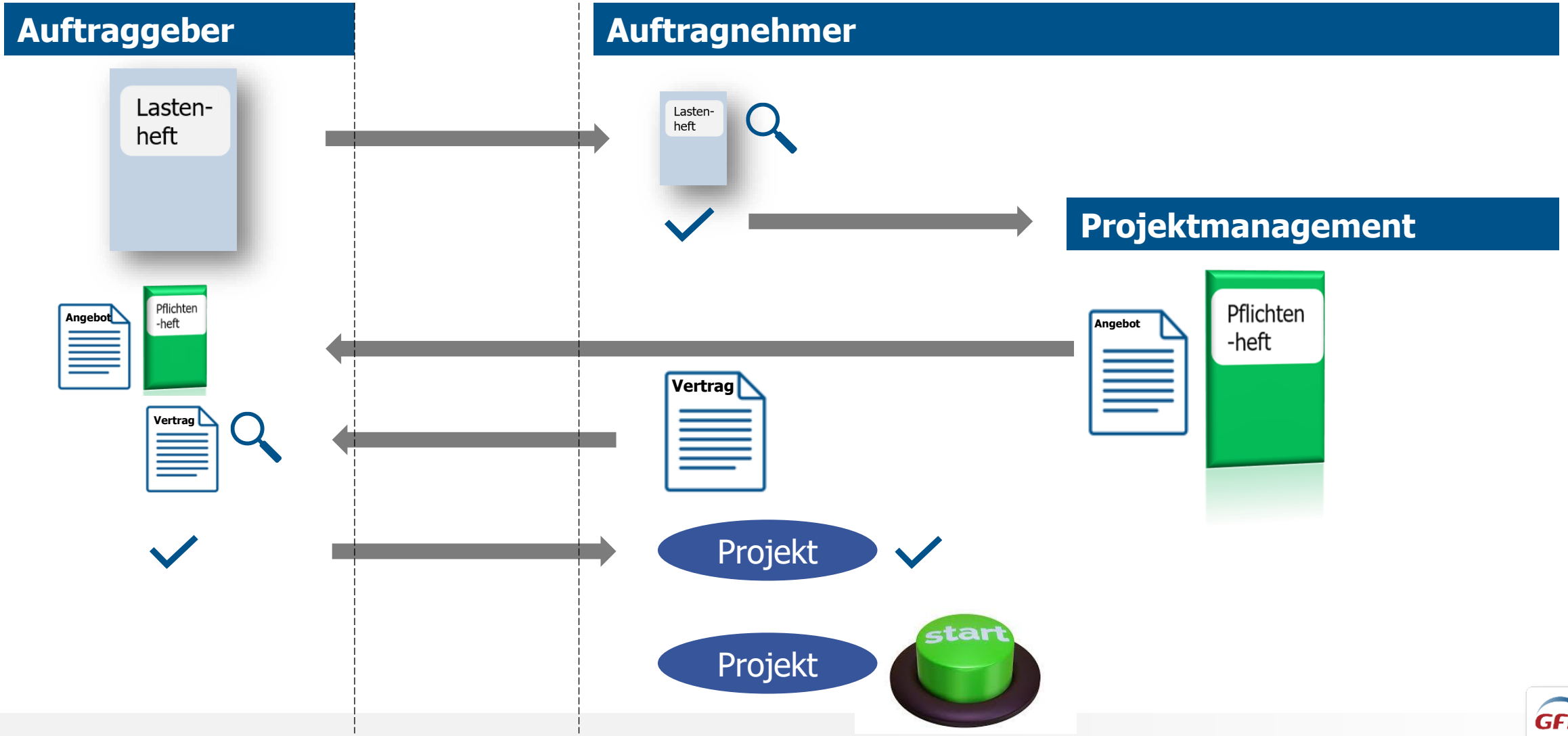
Pflichtenheft

1. Zielbestimmung
   1. Muss-Kriterien
   2. Wunschkriterien
   3. Abgrenzungskriterien
2. Produkteinsatz
   1. Anwendungsbereiche
   2. Zielgruppen
   3. Betriebsbedingungen
3. Produktbedingungen
   1. Software
   2. Hardware
   3. Orgware
   4. Produktschnittstellen

Pflichtenheft

4. Produktfunktionen
5. Produktleistungen
6. Benutzerschnittstellen
7. Qualitätsbestimmungen
8. Globale Testfälle, Referenzen
9. Ergänzungen

Idealtypischer Ablauf bis zum Projektbeginn



Kompetenzcheck

Welche Aussagen sind richtig?

a) Das Lastenheft wird vom Auftraggeber erstellt.

b) Das Pflichtenheft wird vom Auftragnehmer erstellt.

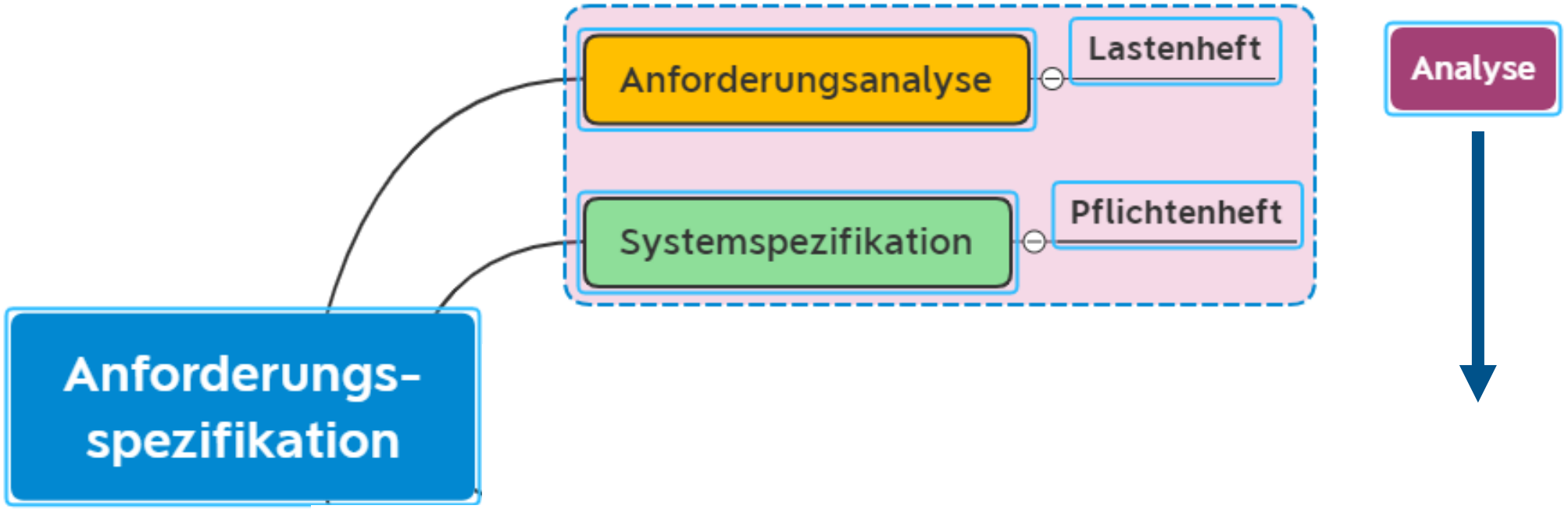
c) Das Lastenheft wird auf Grundlage des Pflichtenheftes erstellt.

d) Nicht funktionale Anforderungen werden nur im Pflichtenheft formuliert.

e) Das Pflichtenheft enthält konkrete Lösungsvorschläge.

f) Das Pflichtenheft ist die Grundlage für den Vertrag zwischen dem Auftraggeber und dem Auftragnehmer.

5.4.3 Den Entwurfsprozess beschreiben



Kompetenzcheck

Welche Aussagen sind richtig?

a) Beim Entwurf einer Software wird u. a. die Architektur der Software festgelegt.

b) Mit dem Design der Software kann schon vor der Analysephase begonnen werden.

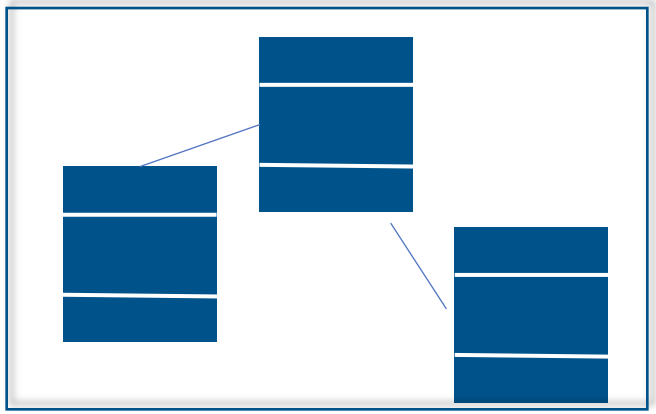
c) Die Entwicklung von Algorithmen gehört zum Detailentwurf.

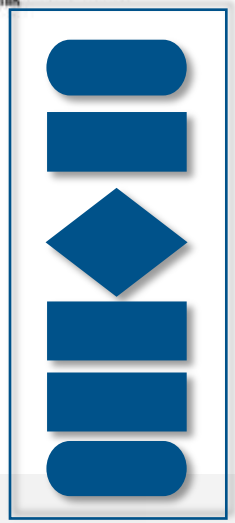
d) Beim Softwareentwurf kommen Modellierungssprachen zum Einsatz.

5.4.4 Modellierungssprachen unterscheiden

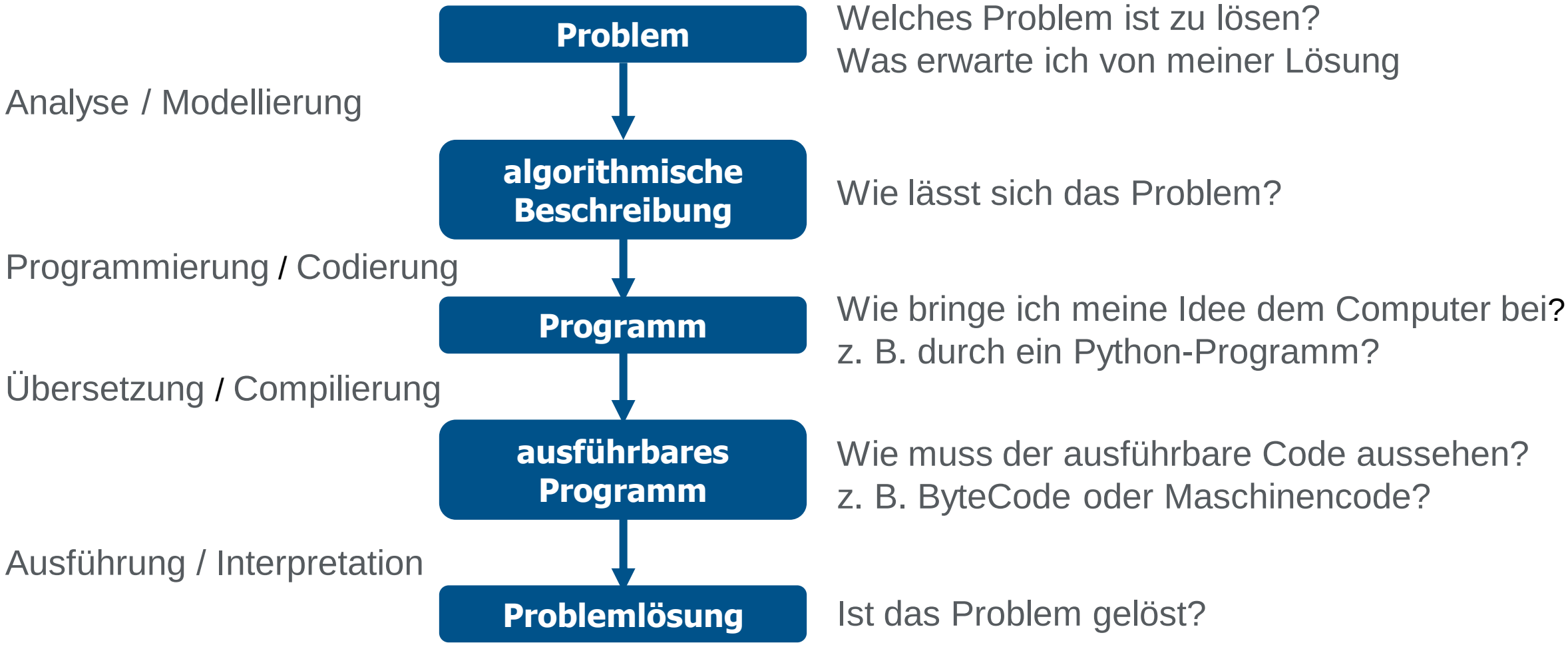
Auftraggeber

Auftragnehmer

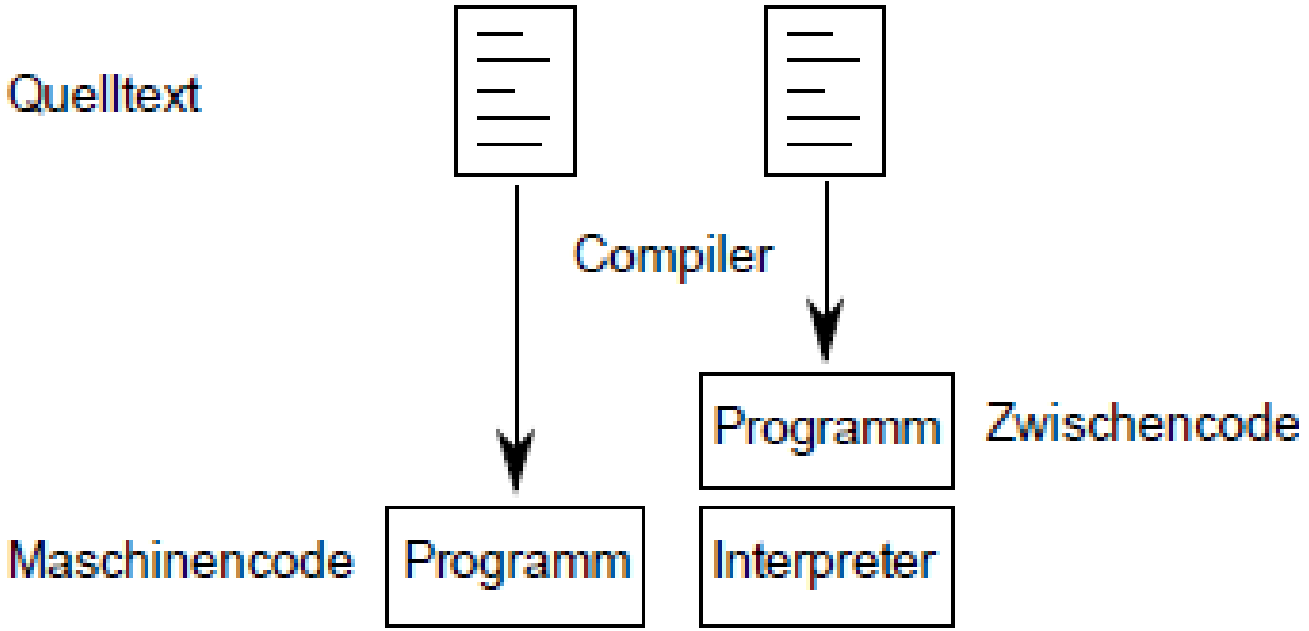




5.4.4 Modellierungssprachen unterscheiden



5.4.4 Modellierungssprachen unterscheiden



Vom Quelltext zum fertigen Programm:
Je nach Programmiersprache wird der Quelltext vom Compiler direkt in Maschinencode übersetzt oder zunächst in einen Zwischencode übersetzt, welcher dann von einem Interpreter ausgeführt wird

5.4.4 Modellierungssprachen unterscheiden

- ist eine eindeutige Handlungsvorschrift zur Lösung eines Problems oder einer Klasse von Problemen
- besteht aus endlich vielen, wohldefinierten Einzelschritten
- Überführt eine bestimmte Eingabe in eine bestimmte Ausgabe

- liefert bei denselben Voraussetzungen das gleiche Ergebnis (Determiniertheit)
- die nächste anzuwendende Regel im Verfahren ist zu jedem Zeitpunkt eindeutig definiert (Determinismus)

5.4.4 Modellierungssprachen unterscheiden

Wer macht hier was falsch?

5.4.4 Modellierungssprachen unterscheiden

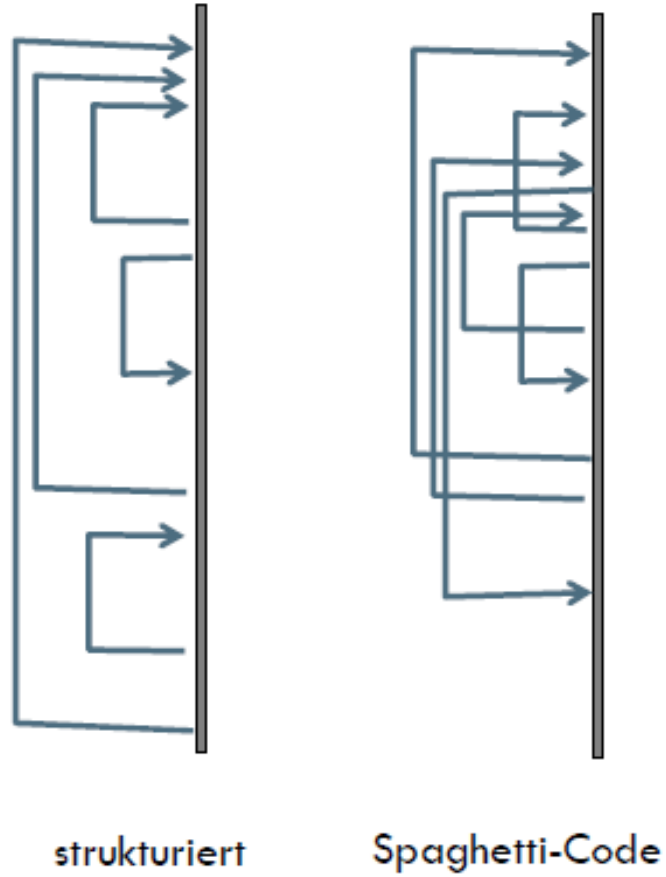
- Programmablaufplan – PAP
- Struktogramm
- Pseudocode
- Datenflussplan
- Entscheidungstabellen
- Unified Modeling Language –UML
- Entity Relationship Model - ERM

5.4.4 Modellierungssprachen unterscheiden

Modellierungssprachen Warum?

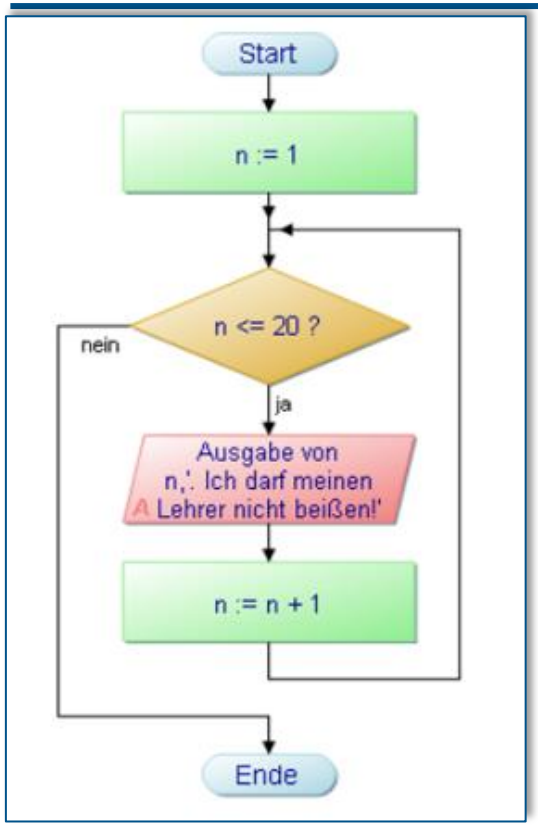
- Übersichtlichkeit wichtig!
- Einschränkungen
  - Schleifen der Programmablaufpläne sind höchstens ineinander geschachtelt
  - Schleifen überkreuzen sich nicht!
- Keine beliebigen Sprünge!

→ Strukturiertes Programmieren



5.4.4 Modellierungssprachen unterscheiden

Programmablaufplan



Struktogramm

berechneFahrtkosten

Parameter:
fahrzeugtyp, tarifart, gefahreneStrecke, startzeit, endezeit

Rückgabewert:
gesamtkosten

float km-preis=0
float gesamtkosten=0
float stundenpreis=0
float km-kosten=0
float zeitkosten=0

Fahrzeugtyp == ?		
1=Kleinwagen	2=Kombi	3=Transporter
km-preis = 0,22	km-preis=0,26	km-preis=0,33
stundenpreis=2,20	stundenpreis=2,80	stundenpreis=4,20

benutzteZeit=berechneStunden(Anfangszeit, Endezeit)

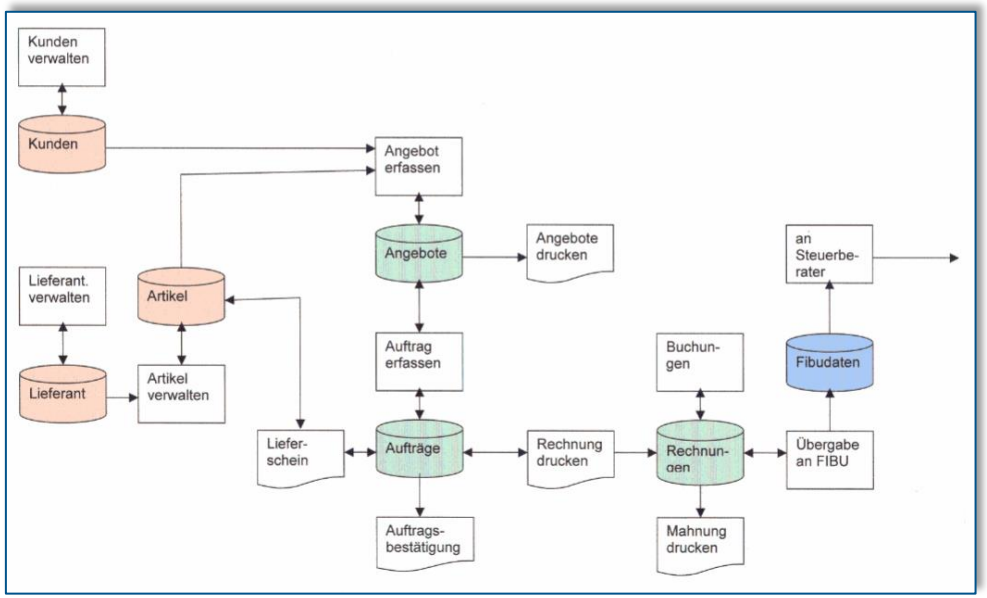
km-kosten=km-preis*gefahreneStrecke
zeitkosten=benutzteZeit*stundenpreis
gesamtkosten=km-kosten+zeitkosten

Tarif==1?	
T	F
∅	gesamtkosten=gesamtkosten-20%

Rückgabe gesamtkosten

5.4.4 Modellierungssprachen unterscheiden

Datenflussplan

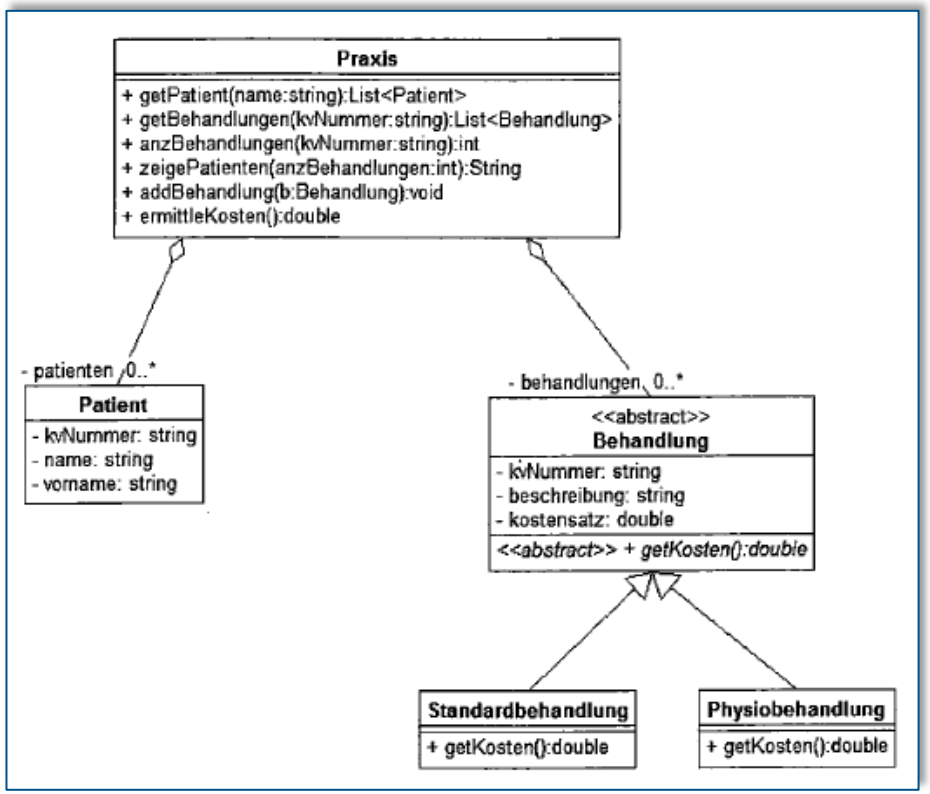


Entscheidungstabellen

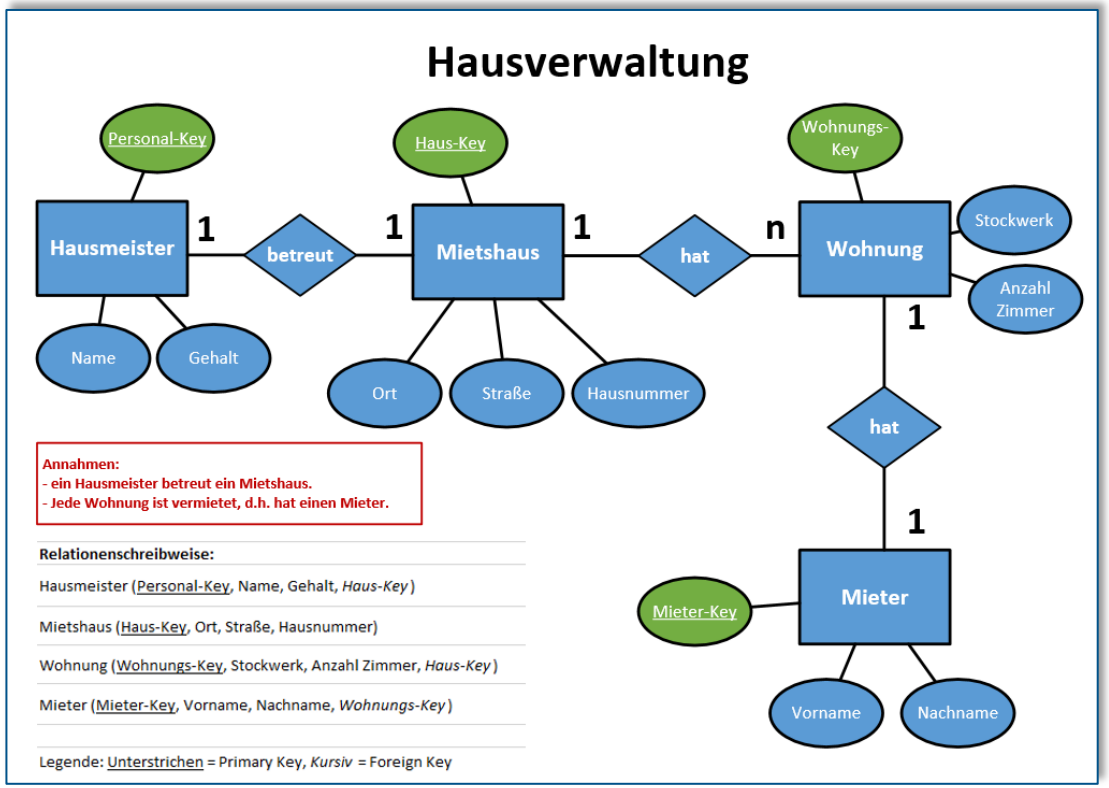
Entscheidungs-tabelle	Regeln			
Bedingungen	R1	R2	R3	R4
Es regnet	J	J	N	N
Es ist kalt	J	N	J	N
Aktionen				
Regenschirm mitnehmen	X	X	–	–
Jacke anziehen	X	–	X	–

5.4.4 Modellierungssprachen unterscheiden

Unified Modeling Language (UML)



Entity Relationship Modell (ERM)

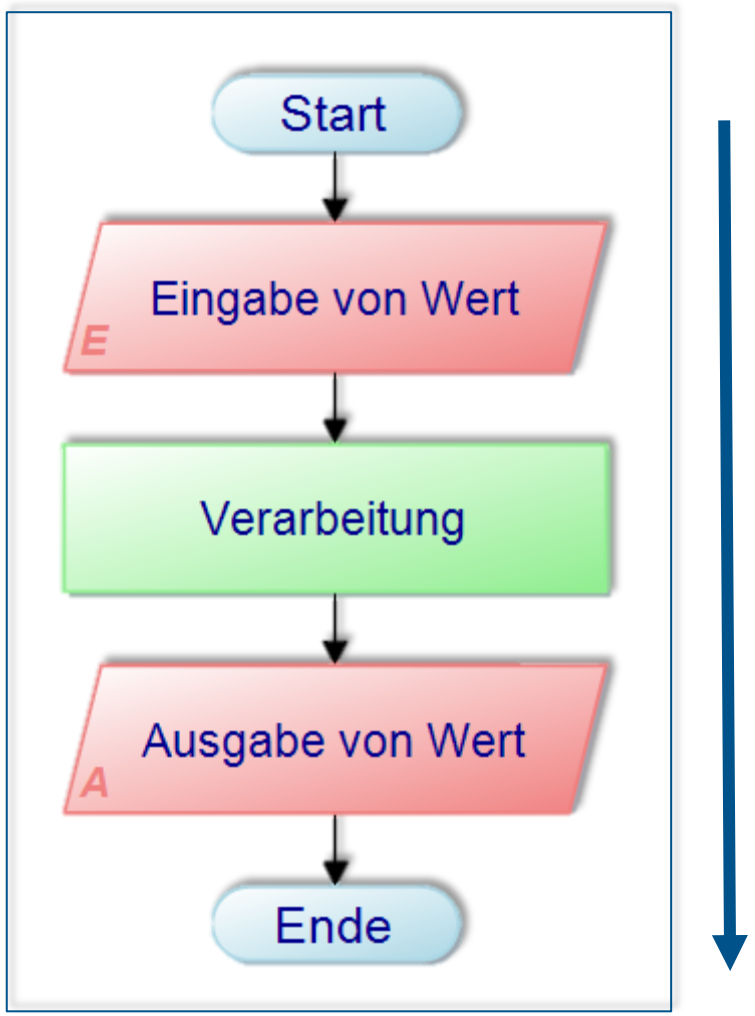


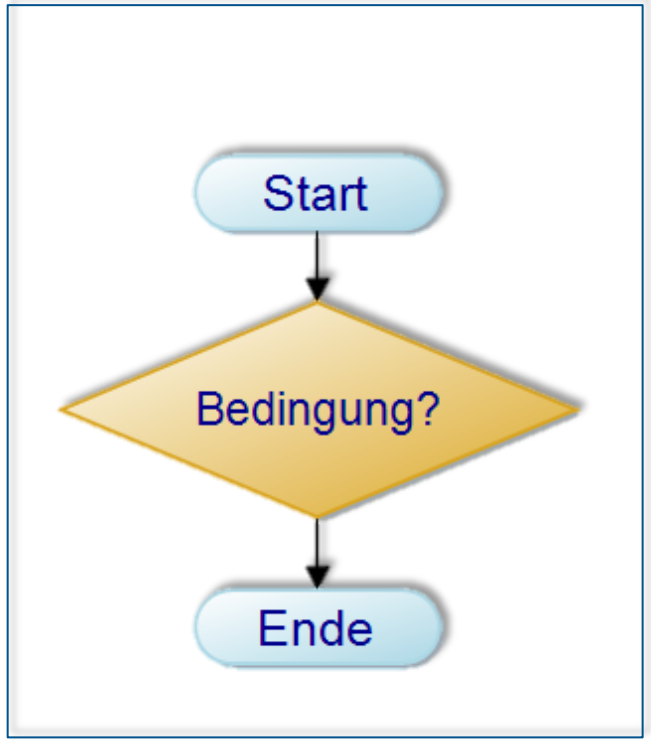
5.4.4 (1) Programmablaufplan - PAP

Ablaufdiagramm für ein Computerprogramm, das auch
als *Flussdiagramm* (*flowchart*) oder *Programmstrukturplan* bezeichnet wird

Es ist eine grafische Darstellung zur Umsetzung eines Algorithmus in einem Programm und beschreibt die Folge von Operationen zur Lösung einer Aufgabe und ist in der DIN 66001 genormt

5.4.4 (1) Programmablaufplan – PAP Eingabe/Verarbeitung/Ausgabe

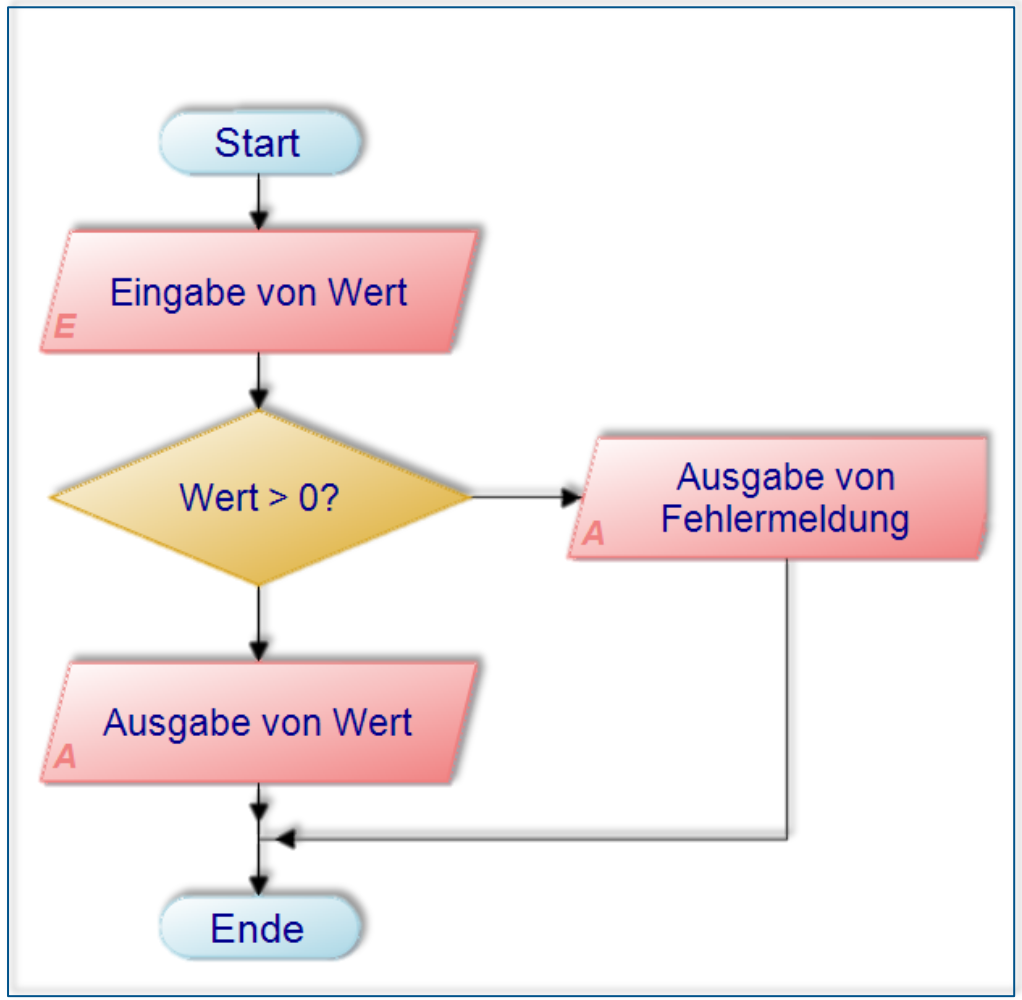


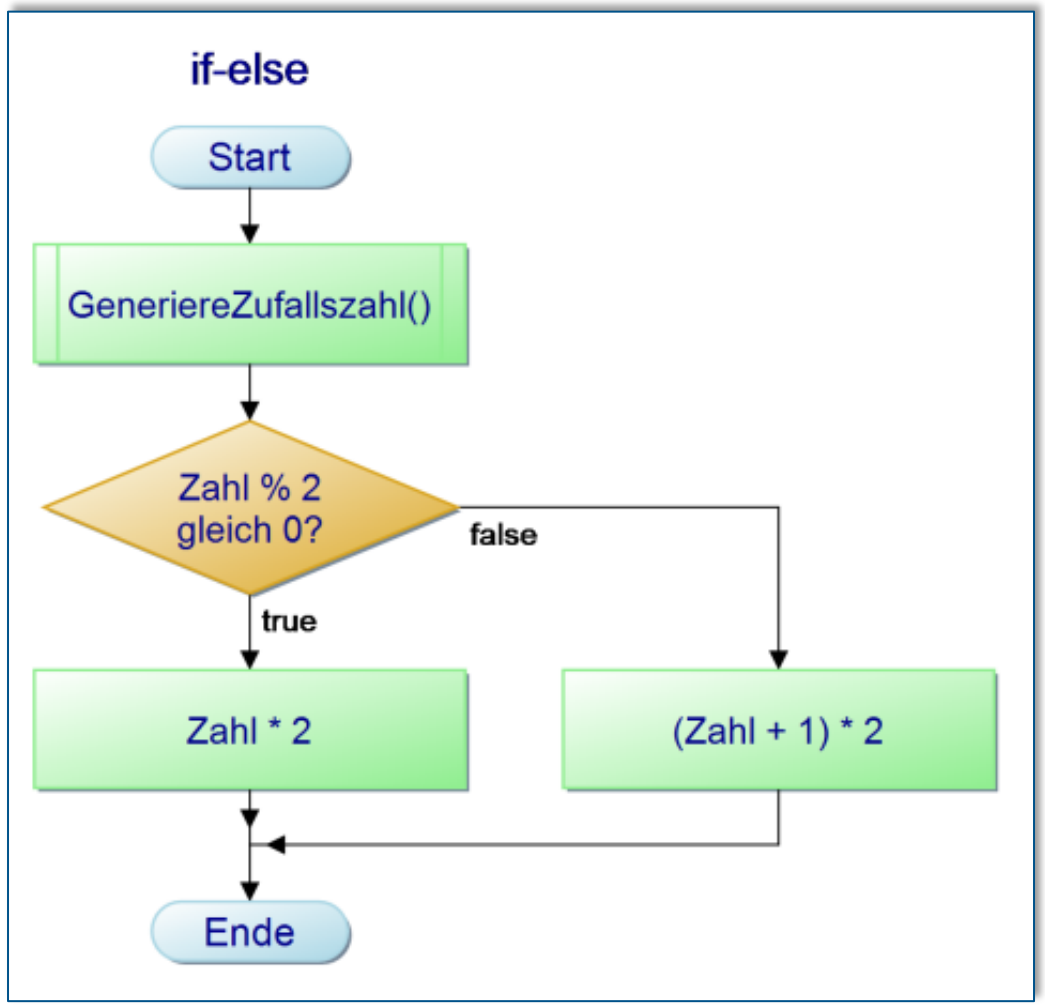


Für

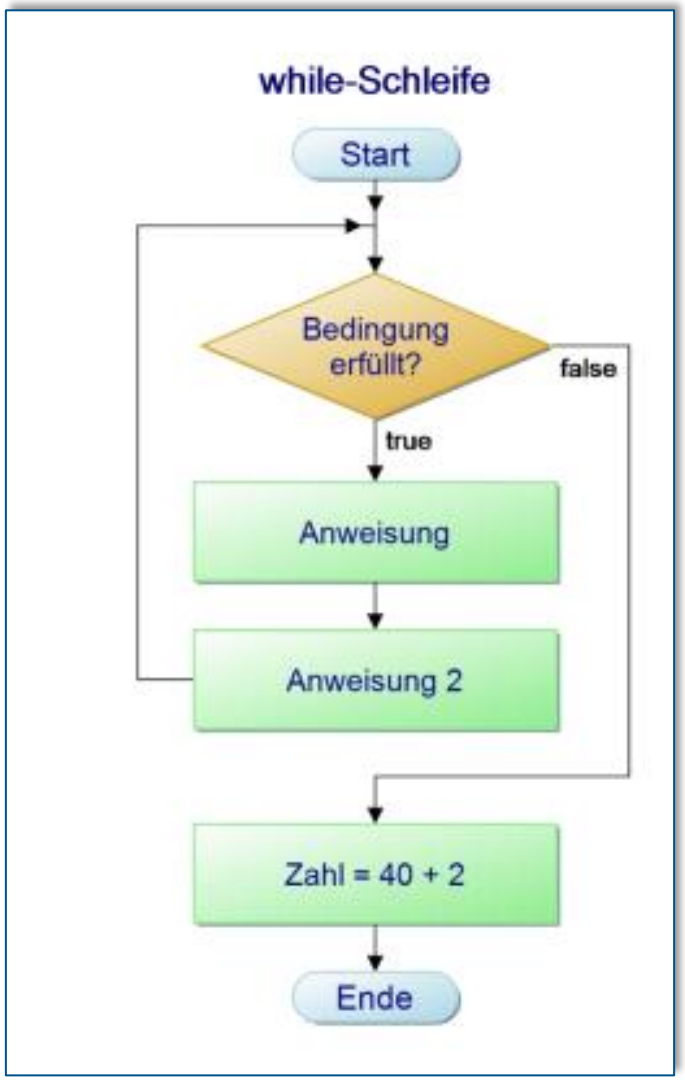
- IF-Anweisungen (Wenn-Dann-Sonst)
- Schleifen (Wiederholungen)

5.4.4 (1) Programmablaufplan – PAP Eingabe/Verarbeitung/Ausgabe



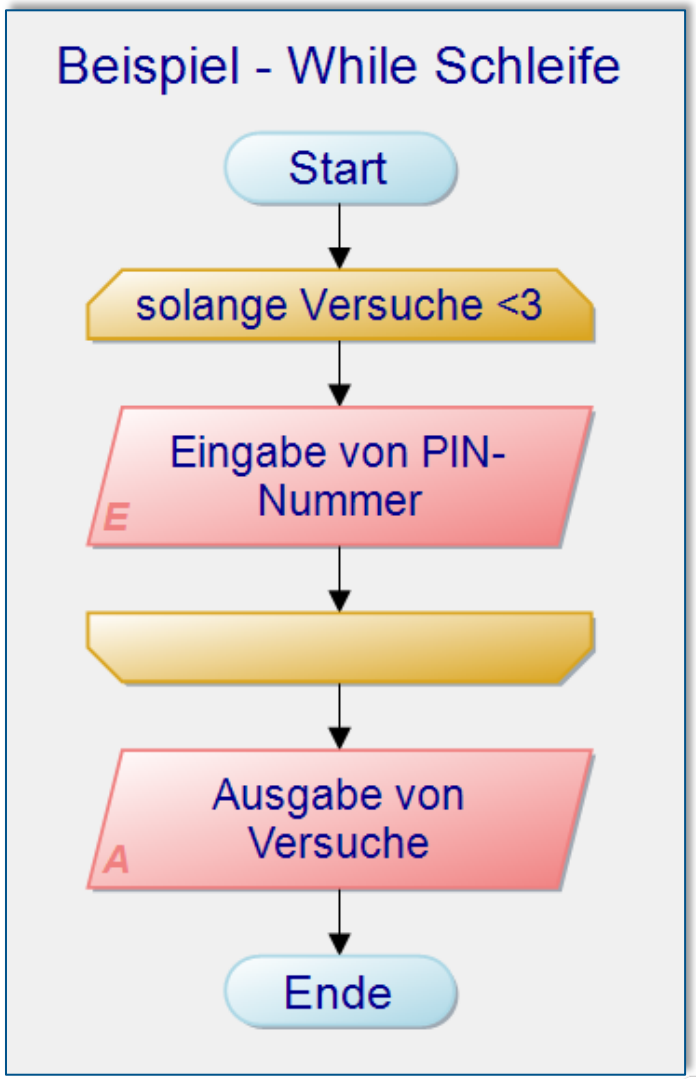


5.4.4 (1) Programmablaufplan – PAP Schleifen

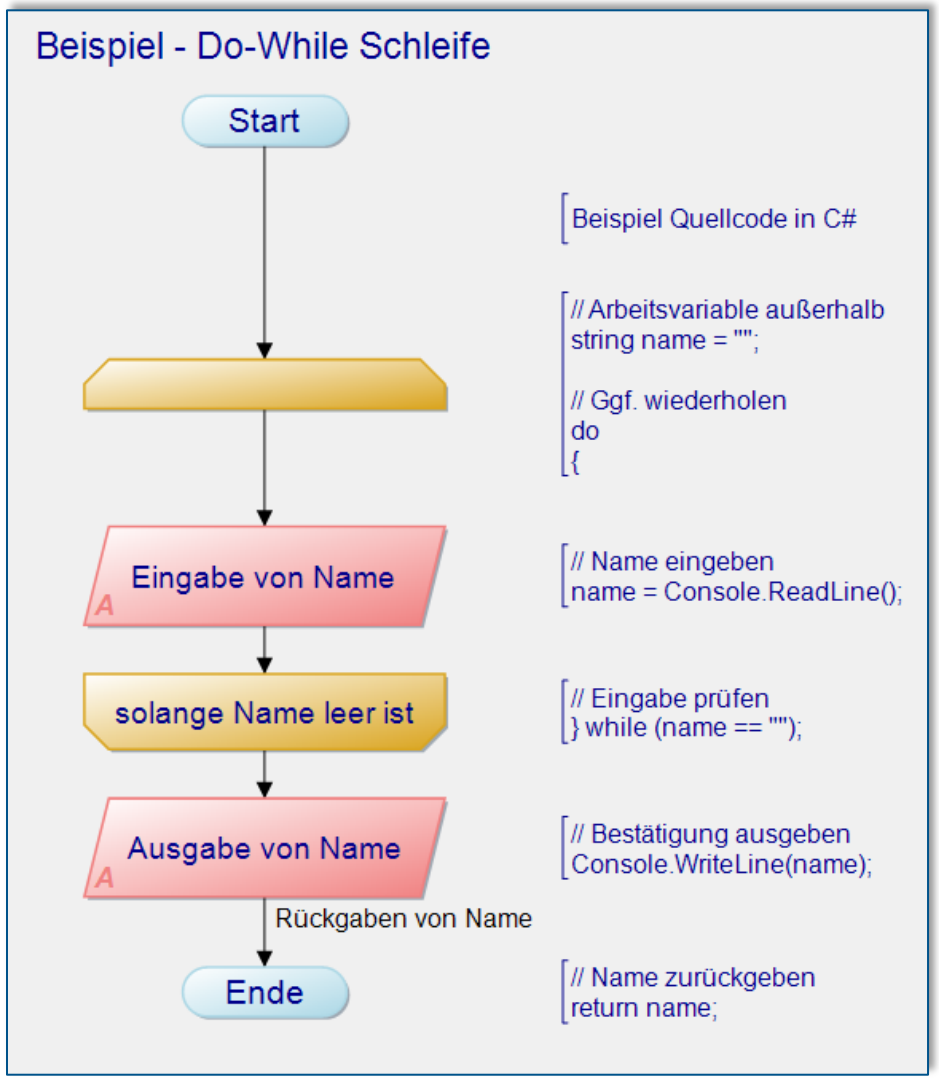


Kopfgesteuerte (Abweisende) Schleife

Führe die Schleife aus, (Betrete die Schleife) solange die Bedingung erfüllt ist



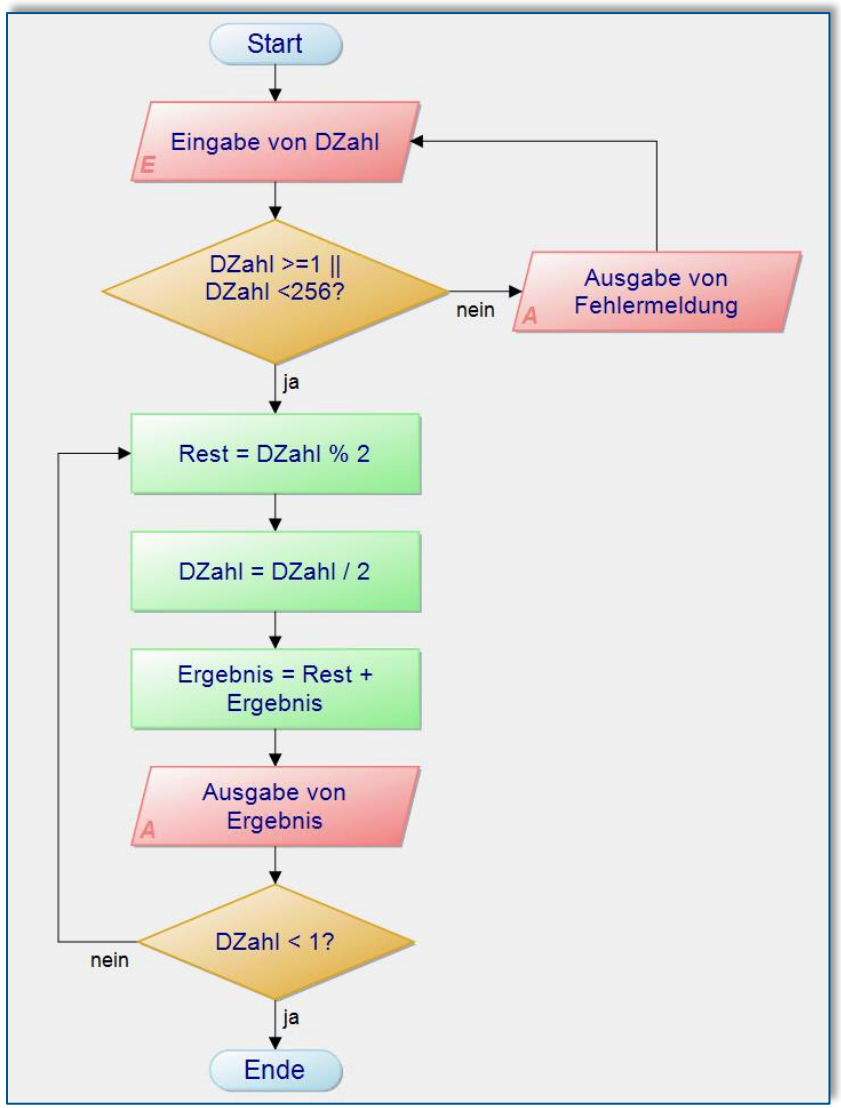
5.4.4 (1) Programmablaufplan – PAP Schleifen



Fußgesteuerte
(akzeptierend; nicht abweisende)
Schleife

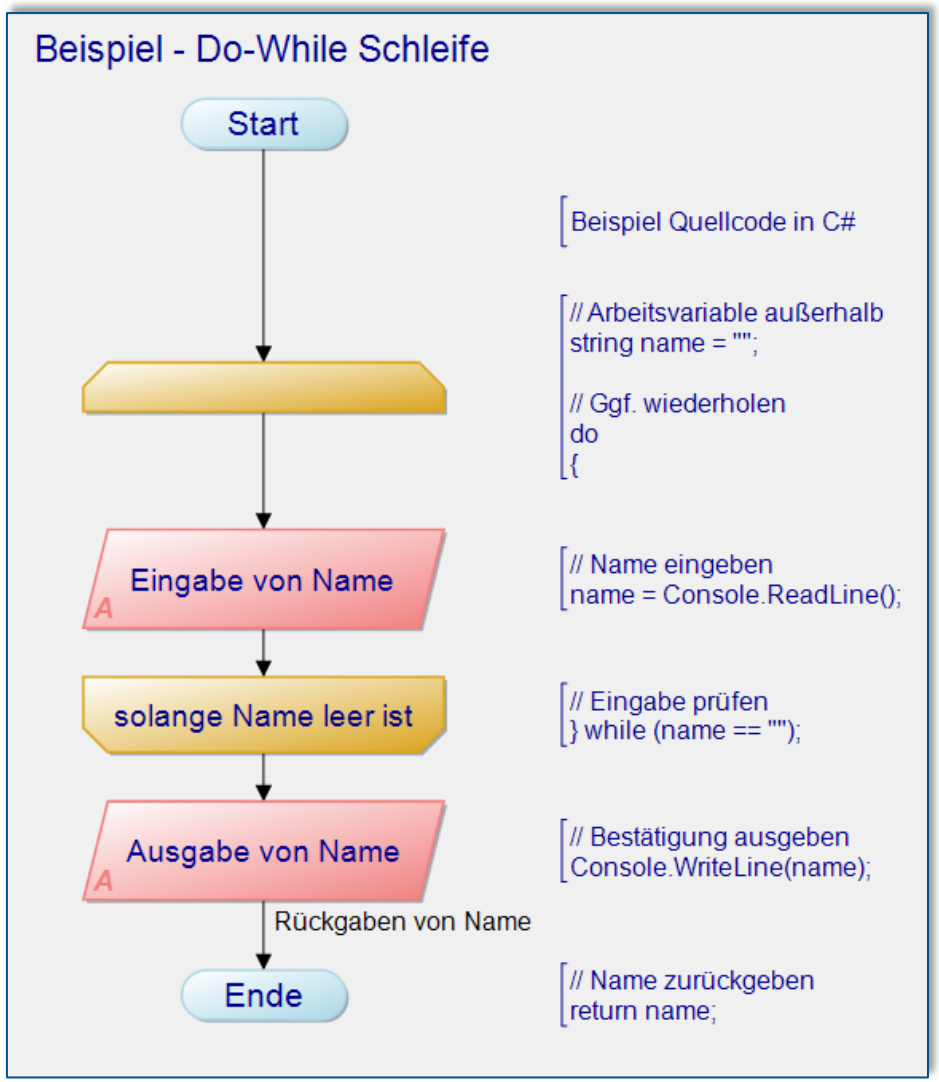
Wiederhole solange Bedingung erfüllt ist

5.4.4 (1) Programmablaufplan – PAP Schleifen



Fußgesteuerte (akzeptierend; nicht abweisende) Schleife

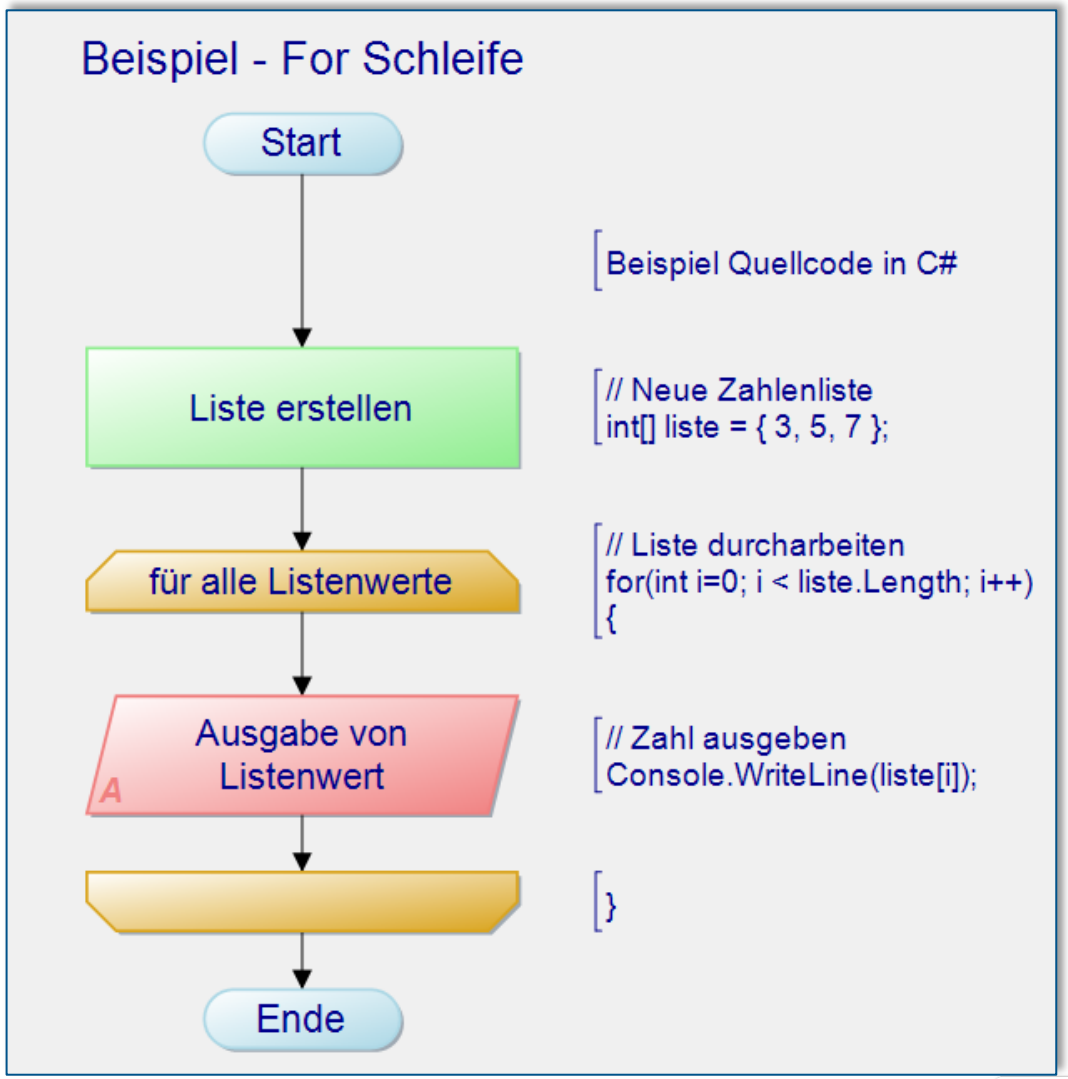
Wiederhole solange Bedingung erfüllt ist



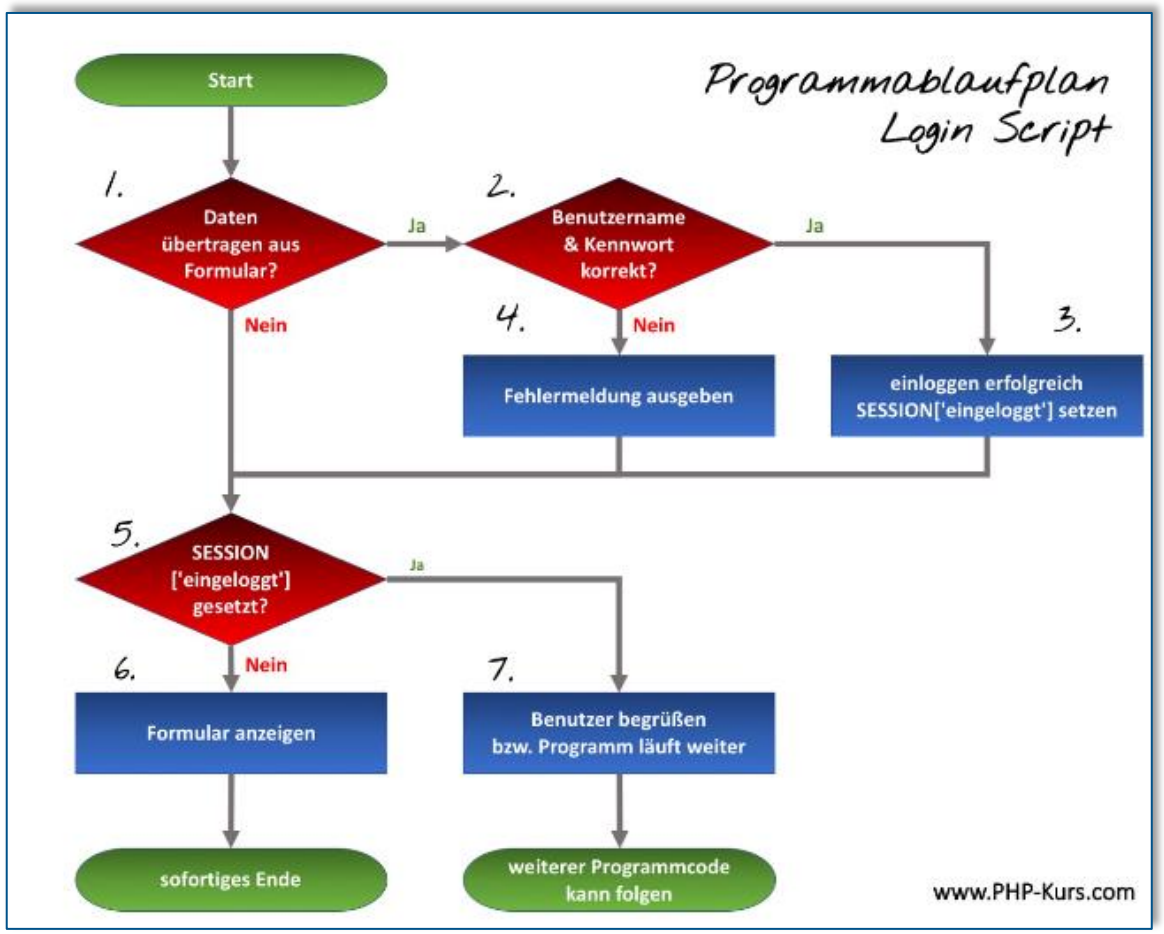
5.4.4 (1) Programmablaufplan – PAP Schleifen

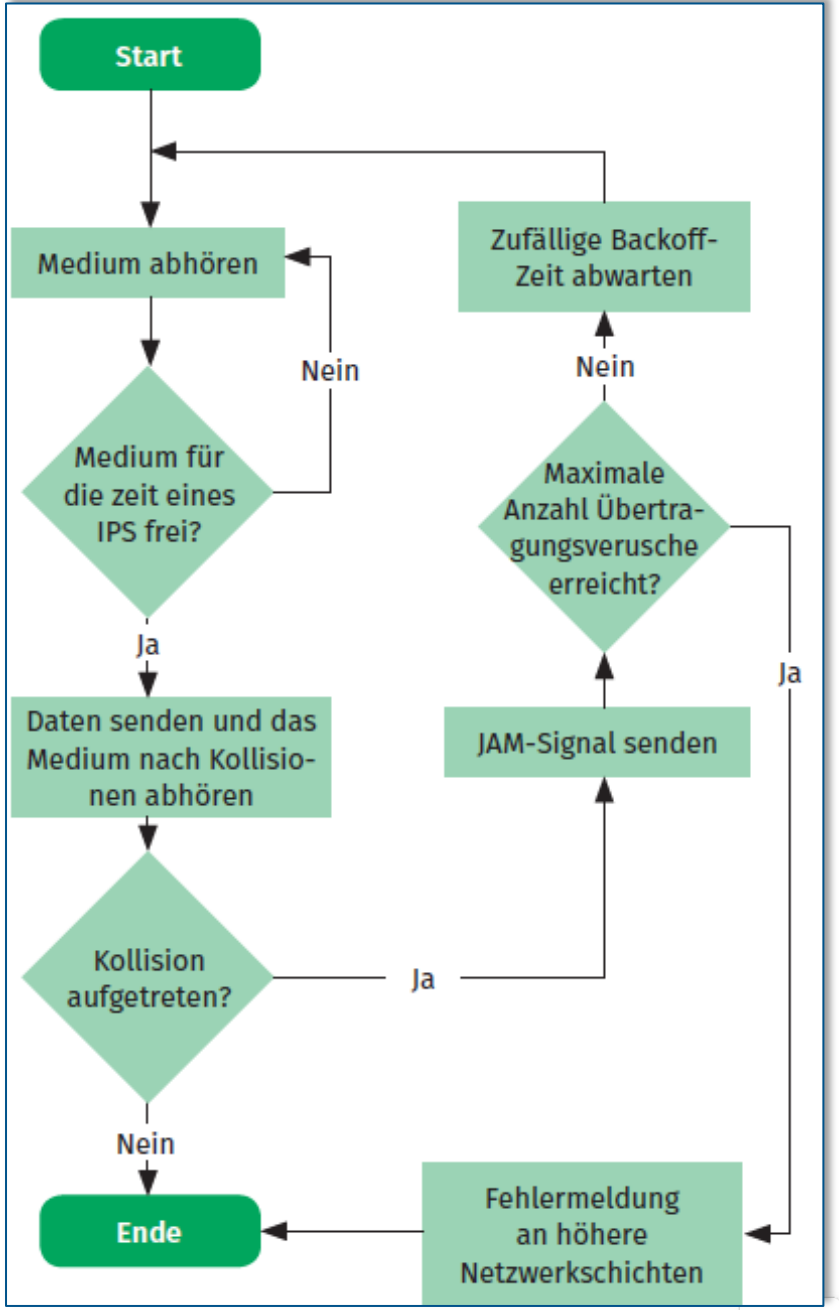
Zählschleife

Führe die Anweisungen in der Schleifen n-mal aus



5.4.4 (1) Programmablaufplan Beispiele





Quelle: aus Westermann LF03 Medienzugriff regeln

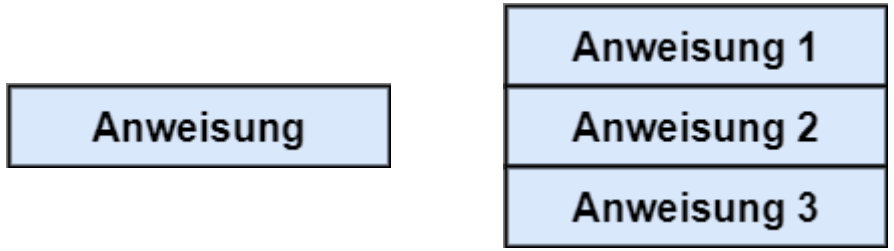
5.4.4 (1) Struktogramme

Diagrammtyp zur Darstellung von Programmentwürfen im Rahmen der Methode der strukturierten Programmierung

Es wurde 1972/73 von Isaac Nassi und Ben Shneidermann entwickelt und ist in der DIN 66261 genormt

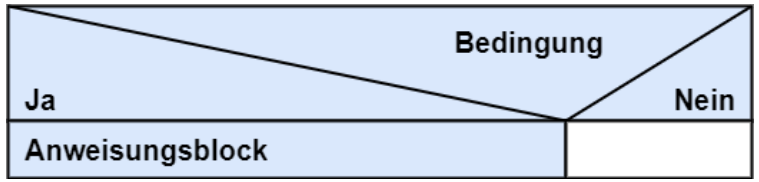
5.4.4 (1) Struktogramme

Darstellung von Anweisungen,
Unterprogrammen
Folge(Sequenz)

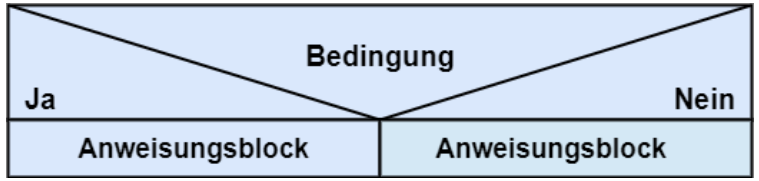


Auswahl/Alternative

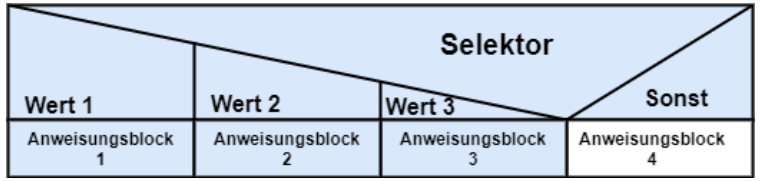
Darstellung von Verzweigung, (einseitig)



Darstellung von Verzweigung, (zweiseitig)

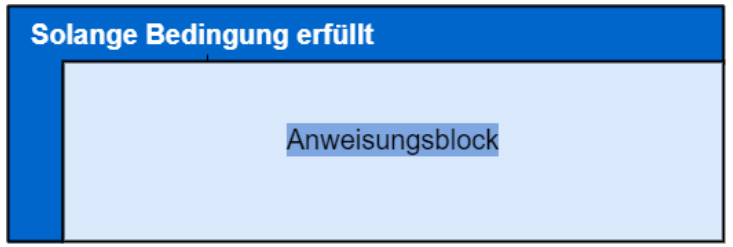


Darstellung einer Mehrfachauswahl

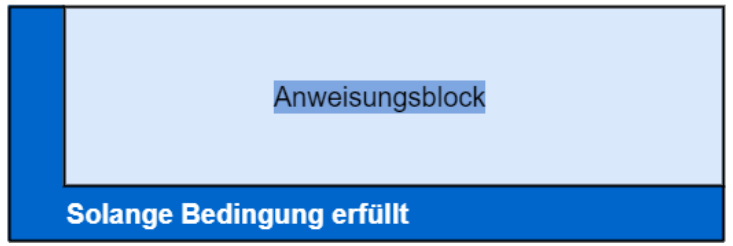


5.4.4 (1) Struktogramme

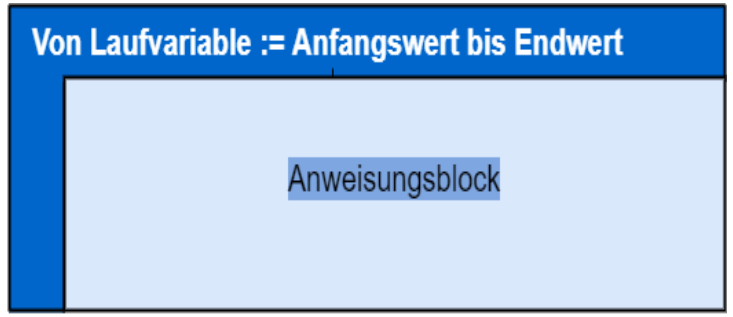
Wiederholungsstruktur mit Anfangsbedingung
Der Anweisungsblock wird so lange durchlaufen, wie die Bedingung zutrifft



Wiederholungsstruktur mit Endebedingung
der Anweisungsblock wird hier mindestens einmal durchlaufen, weil die Bedingungsprüfung erst im Anschluss an den Anweisungsblock stattfindet



Wiederholungsstruktur mit Zählvariable
Die Anzahl der Schleifendurchläufe wird durch eine Zählvariable festgelegt. Im Schleifenkopf werden der Startwert der Zähl-variablen, der Endwert und die Veränderung der Zählvariablen nach jedem Schleifendurchlauf angegeben



5.4.4 (2) Entscheidungstabellen

Die DIN 66241 nennt vier Teile einer Entscheidungstabelle

- Die für die Entscheidungssituation relevanten Bedingungen werden im Bedingungsteil erfasst
- Die Beschreibung der möglichen Aktionen erfolgt im Aktionsteil
- Die Bedingungsanzeige enthält die möglichen Kombinationen von Bedingungen (also Bedingung wird erfüllt / wird nicht erfüllt / ist irrelevant)
- Die Aktionsanzeige enthält die möglichen Aktionen in Abhängigkeit von bestimmten Bedingungen (Aktion ausführen / nicht ausführen)

5.4.4 (2) Entscheidungstabellen

Entscheidungs-tabelle	Regeln							
Bedingungen	R1	R2	R3	R4	R5	R6	R7	R8
Bedingung 1	J	J	J	J	N	N	N	N
Bedingung 2	J	J	N	N	J	J	N	N
Bedingung 3	J	N	J	N	J	N	J	N
Aktionen								
Aktion 1	-	X	-	-	X	-	-	X
Aktion 2	-	-	X	-	X	X	X	-
Aktion 3	-	-	-	X	X	X	-	X
...								

5.4.4 (2) Entscheidungstabellen

		Regeln		
		Regel 1	Regel 2	Regel 3
Bedingungen	Lieferung durch Auftraggeber angenommen	Ja	Nein	Nein
	Lieferung durch Auftraggeber angenommen	Ja	Ja	Nein
	Lieferung durch Auftraggeber angenommen	Nein	Nein	Nein
Aktionen	Erste Erinnerung verschickt	x	-	-
	Erste Mahnung vorbereiten	x	-	-
	Telefonischen Austausch suchen	-	x	x
	Intern eskalieren	-	x	-
	Auftraggeber in ABC-Analyse abstufen	-	-	-

5.4.4 (3) Pseudocode

- von griech. pseudo = unecht
- Ist ein nicht funktionärer Code, den man schreibt, um eine strukturierte Übersicht über ein Programm und seine Aktionen zu erhalten
- Wird vor allem bei höheren Programmiersprachen als Hilfestütze verwendet, kann bei komplexen Projekten von Nutzen sein

5.4.4 (3) Pseudocode

- Pseudocode ist eine Mischung aus natürlicher Sprache und einer höheren Programmiersprache

- Eine Beschreibung eines Algorithmus in Pseudocode ist einerseits exakter als eine Beschreibung in natürlicher Sprache, andererseits aber noch nicht so detailliert wie eine Implementation als Computerprogramm

- Oft wird Stepwise Refinement angewendet, d. h. der zuerst noch sehr kurze und wenig formale Pseudocode wird in mehreren Schritten verfeinert, bis am Schluss der Schritt zum Computerprogramm nur noch klein ist

Pseudocode ist subjektiv und kein Standard!

- Es gibt keinen festgelegten Satzbau, den man für Pseudocode unbedingt benutzen muss
- Es ist aber übliche Berufsethik, Pseudocode-Standardstrukturen zu verwenden, die andere Programmierer leicht verstehen können
- Falls ein Projekt allein codiert wird, ist das Wichtigste, dass Pseudocode dabei hilft, die Gedanken zu strukturieren und den Plan umzusetzen
- Falls man mit anderen zusammen an einem Projekt arbeitet – ob sie nun Fachkollegen, untergeordnete Programmierer oder nicht-technische Mitarbeiter sind, ist es wichtig, zumindest einige Standardstrukturen zu verwenden. So kann jeder die Absicht leicht verstehen

5.4.4 (3) Pseudocode

Ein Pseudocode für ein Quiz-Spiel könnte beispielweise folgender sein:

```
Wenn Programm beginnt
wiederhole 10 mal {
    stelle eine Zufallsfrage
    warte auf Antwort
    falls Antwort richtig dann {
        bestätige Antwort
        gebe einen Punkt
    } ansonsten {
        nenne richtige Antwort
    }
}
beende das Programm
```

5.4.4 (3) Pseudocode

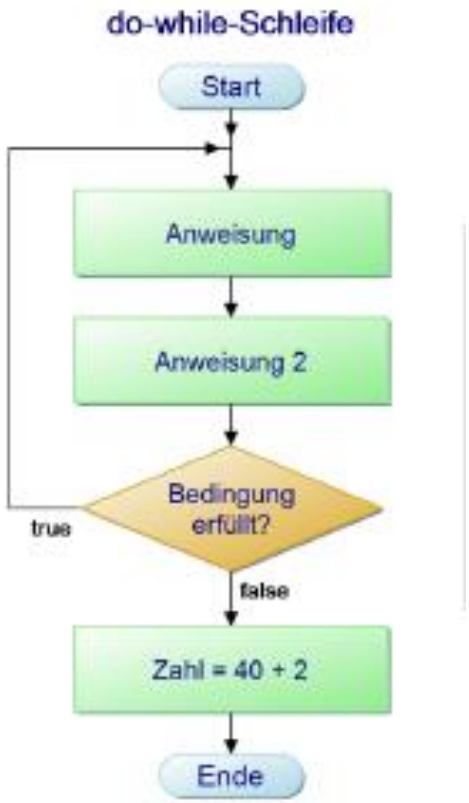
Kopfgesteuerte Schleife


```
while (Bedingung)
        // Anweisung 1
        // …
        // Anweisung n
Zahl = 40 + 2
```

```
16  while(Bedingung) {
17      // Anweisung 1
18      // ...
19      // Anweisung n
20  }
21  Zahl = 40+2;
```

5.4.4 (3) Pseudocode

Fußgesteuerte Schleife

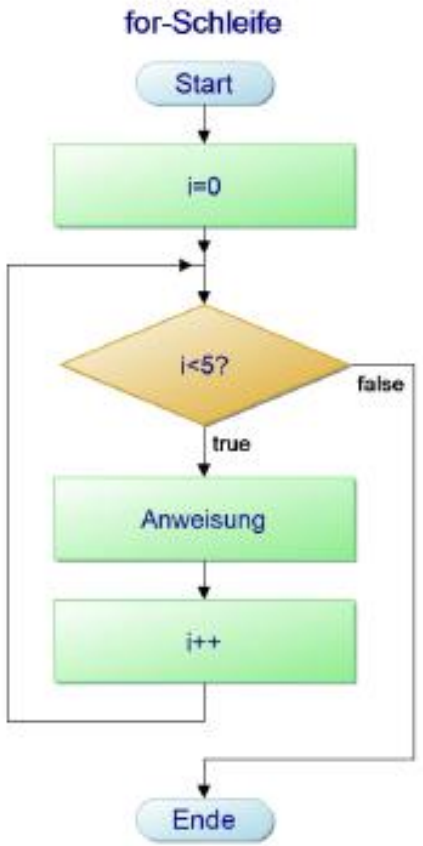


Führe aus
// Anweisung 1
// …
// Anweisung n
While(bedingung)
Zahl = 40 + 2

```
25  do {
26      // Anweisung 1
27      // ...
28      // Anweisung n
29  } while(Bedingung);
30  Zahl = 40+2;
```

5.4.4 (3) Pseudocode

Zähl-Schleife




```
i = 0
for i = 0 to 5
        i = i ++
        // Anweisungen
i=0
```

```
64  for(i=0;i<5;i++)
65  {
66      // Anweisungen
67  }
68
69  i=0;
70  while(i<5)
71  {
72      // Anweisungen
73      i++;
74  }
```

5.4.4 (3) Pseudocode

Aufgabe

Situation zu den Teilaufgaben 3.9 und 3.10

Die rackwatch AG entwickelt eine Temperaturüberwachung für Serverräume. Eine Ampelanzeige vor dem Serverraum soll die Temperaturzustän visuell darstellen.

Folgende Spezifikation wurde dafür festgelegt:

Farbe der Anzeige	Bedingung
grün	Alle Sensoren liefern Messwerte zwischen 20° C und 30° C.
gelb	Mindestens ein Sensor liefert einen Messwert, der unter 20° C oder der zwischen 30° C und 35° C liegt.
rot	Mindestens ein Sensor liefert einen Messwert, der über 35° C liegt. oder Mindestens ein Sensor ist nicht erreichbar.

Ein Kollege hat bereits folgenden Pseudocode erstellt:

Zeile	Pseudocode
1	farbe=grün
2	**Schleife** über alle Sensoren
3	**wenn** farbe==grün
4	**wenn** der Sensor erreichbar ist
5	**wenn** sensor.lese_temp()<20
6	farbe=gelb
7	**wenn** sensor.lese_temp()>30
8	farbe=gelb
9	**wenn** sensor.lese_temp()>35
10	farbe=rot
11	**sonst**
12	farbe=rot
13	**Ende der Schleife** über alle Sensoren
14	Lasse die Anzeige in „farbe" leuchten

Testdaten

Situation Nr.	Sensor 1	Sensor 2	Sensor 3	Erwartete Farbe
1	21° C	26° C	27° C	grün
2	23° C	34° C	27° C	gelb
3	25° C	nicht erreichbar	27° C	rot
4	27° C	33° C	40° C	rot
5	29° C	26° C	27° C	grün
6	31° C	nicht erreichbar	nicht erreichbar	rot

5.4.4 (3) Pseudocode

Aufgabe

Sie sollen mit einem Schreibtischtest den Pseudocode prüfen.

Dazu stehen Ihnen die nachstehenden Testdaten mit den Situationen (Nr. 1 bis 6) zur Verfügung. In zwei Situationen liefert der Pseudocode nicht das geforderte Ergebnis.

Ermitteln Sie die Situationen, in denen der Pseudocode das geforderte Ergebnis **nicht** liefert.

Tragen Sie die Nummern der **zwei** zutreffenden Situationen in die Kästchen ein.

Testdaten

Situation Nr.	Sensor 1	Sensor 2	Sensor 3	Erwartete Farbe
1	21° C	26° C	27° C	grün
2	23° C	34° C	27° C	gelb
3	25° C	nicht erreichbar	27° C	rot
4	27° C	33° C	40° C	rot
5	29° C	26° C	27° C	grün
6	31° C	nicht erreichbar	nicht erreichbar	rot

5.4.4 (3) Pseudocode

Aufgabe

- Mit welcher der folgenden Änderungen im Pseudocode kann der Fehler beseitigt werden ?
- Kreuzen Sie die richtige Ziffer an.

1 Die Zeile 1 und 2 vertauschen
2 Die Zeile 3 durch „wenn farbe != rot“ ersetzen
3 Die Zeile 4 durch „wenn der Sensor nicht erreichbar ist“ ersetzen
4 Die Zeilen 11 und 12 löschen
5 Die Zeilen 13 und 14 vertauschen

5.4.4 (4) Unified Modeling Language - UML

- UML ist eine grafische Modellierungssprache, welche zur Planung von objektorientierter Software eingesetzt wird
- Sie ist heute das am meisten eingesetzte Modellierungsmittel für die Softwaresystemmodellierung

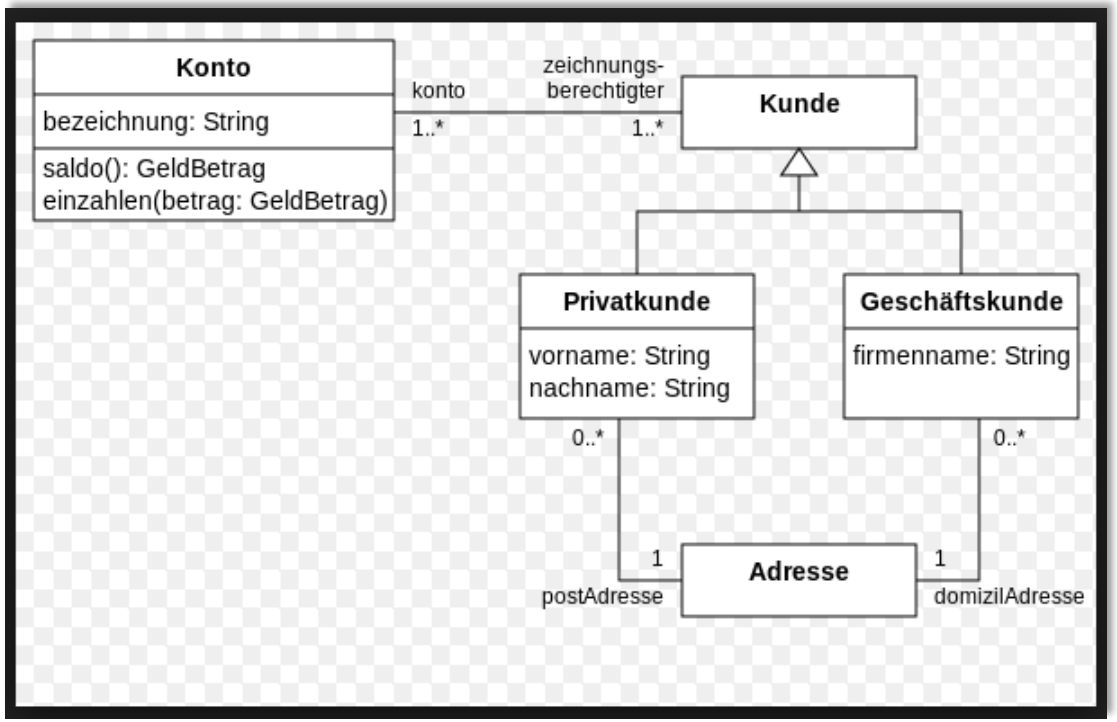
- Sie wird von der Object Management Group (OMG) entwickelt und ist in der ISO/IEC 19505 genormt

5.4.4 (4) Unified Modeling Language - UML

- Die Unified Modeling Language (vereinheitlichte Modellierungssprache), kurz **UML**, ist eine grafische Modellierungssprache zur Spezifikation, Konstruktion und Dokumentation von Software-Teilen und anderen Systemen

- Kein Vorgehensmodell!

  „Nur“ Sammlung von Beschreibungsmitteln



5.4.4 (4) Unified Modeling Language - UML

- Sie enthält Diagramme und Prosa-Beschreibungsformen
- Mit Hilfe der UML können statische, dynamische und Implementierungsaspekte von Softwaresystemen beschrieben werden
- "Die UML ist damit zur Zeit die umfassendste Sprache und Notation zur Spezifikation, Konstruktion Visualisierung und Dokumentation von Modellen für die Software-entwicklung." (*Gabriele Bannert - Objektorientierter Softwareentwurf mit UML)*
- Da die UML mit dem Anspruch entwickelt wurde, eine universell gültige Notation zur Modellierung von Software-Systemen zur Verfügung zu stellen, beinhaltet sie kein Vorgehensmodell
- Sie definiert keine Regeln zur Anwendung der verschiedenen Beschreibungselemente. Sie bildet mit ihren Beschreibungselementen die Basis verschiedener Vorgehensmodelle

5.4.4 (4) Unified Modeling Language - UML

- Weil die Methoden von Booch, Rumbaugh und Jacobson (Drei Amigos) bereits sehr populär waren und einen hohen Marktanteil hatten, bildete die Zusammenführung zur Unified Modeling Language (UML) einen Quasi-Standard

- Schließlich wurde 1997 die UML in der Version 1.1 bei der Object Management Group (OMG) zur Standardisierung eingereicht und akzeptiert

- Die Versionen 1.2 bis 1.5 enthalten jeweils einige Korrekturen. Die Version 2.0 ist bei der OMG in Vorbereitung und wird ca. 2003/2004 erscheinen. Aktuelle Informationen hierzu finden Sie unter http://www.omg.org/uml/

5.4.4 (4) Unified Modeling Language - UML

Verhaltensdiagramme (dynamische Aspekte)

- Aktivitätsdiagramm - Activity Diagram
- Anwendungsfalldiagramm - Use Case Diagram
- Zustandsdiagramm - State Machine Diagram

- Interaktionsdiagramme - Interaction Diagram
  - Interaktionsübersichtdiagramm - Interaction Overview Diagram
  - Sequenzdiagramm - Sequence Diagram
  - Kommunikationsdiagramm - Communication Diagram
  - Zeitverlaufsdiagramm - Timing Diagram

5.4.4 (4) Unified Modeling Language - UML

Strukturdiagramme (statische Aspekte)

- Klassendiagramm - Class Diagram
- Objektdiagramm - Object Diagram
- Komponentendiagramm - Component Diagram
- Paket Diagramm - Package Diagram
- Kompositionsstrukturdiagramm - Composite Structure Diagram
- Verteilungsdiagramm - Deployment Diagram

Kompetenzcheck

Welche Aussagen sind richtig?

a) PAP und Struktogramm sind grafische Modellierungssprachen.

b) UML ist eine veraltete grafische Modellierungssprache.

c) Das ER-Modell findet bei der Planung von relationalen Datenbanken Anwendung.

d) Bei Struktogrammen werden die einzelnen Elemente durch Pfeile verbunden.

e) Entscheidungstabellen haben immer drei Bedingungen.

f) Pseudocode ist an die Programmiersprache Java angelehnt.

g) Das Klassen- und das Anwendungsfalldiagramm sind UML-Diagramme.

Kompetenzcheck

- Entwickeln Sie einen Programmablaufplan, welcher das Betanken eines Autos an der Tankstelle darstellt

- Entwickeln Sie ein Struktogramm für folgende Aufgabenstellung:
  Nach Eingabe von drei Zahlen soll der höchste Wert dieser Zahlen ausgeben werden

- Es soll ein Algorithmus entwickelt werden, welcher den Speicherbedarf für digitale Bilder berechnet und in MiB ausgibt. Eingabewerte ist: Anzahl der Bilder, Höhe und Breite eines einzelnen Bildes in Pixeln und die Farbtiefe pro Pixel in Bit. Erstellen Sie für den Algorithmus ein Struktogramm

Kompetenzcheck

- Ein Unternehmen beschließt, seinen Mitarbeitern Aktien zur Vorzugskonditionen anzubieten. Folgende Bezugsberechtigungen wurden festgelegt. Mitarbeiter mit mehr als 10 Jahren Betriebszugehörigkeit können Aktien erwerben. Mitarbeiter, welche mehr als 2 Jahre zum Betrieb gehören, können max. 20 Aktien beziehen. Mitarbeiter, welche außertariflich bezahlt werden oder sich in einem gekündigten Arbeitsverhältnis befinden, haben in keinem Fall einen Anspruch auf den Erwerb der Aktien. Entwerfen Sie eine Entscheidungstabelle.

- Beschreiben Sie mithilfe von Pseudocode einen Algorithmus, welcher das Produkt aller ganzen Zahlen, welche durch 5 und durch 7 teilbar sind, berechnet und ausgibt. Der Zahlenbereich beginnt bei 1 und endet bei einem Endwert, welcher größer als 1 ist und von dem Benutzer eingegeben wird.

Zusammenfassung – Den Prozess der Anforderungsspezifikation

IT-Berufe
Grundstufe 1-5

Westermann
Kapitel 5.4