SortierALGORITHMEN

Def.: Eine Bearbeitungsvorschrift heißt **Algorithmus**, wenn sie folgende Eigenschaften erfüllt:

1. Die Vorschrift ist mit endlichen Mitteln beschreibbar.
2. Sie liefert auf eine eindeutig festgelegte Weise zu einer vorgegebenen Eingabe in endlich vielen Schritten genau eine Ausgabe.

Aus der Definition folgt:

- Die Verfahrensbeschreibung muss an jeder Stelle eindeutig festlegen, welcher Bearbeitungsschritt als nächstfolgender auszuführen ist.
- Jeder Bearbeitungsschritt in der Folge muss auch ausführbar sein.
- Die Verfahrensbeschreibung muss eine endliche Länge besitzen.
- Jeder Bearbeitungsschritt muss festlegen, was zu tun ist und womit (mit welchen Daten) der Schritt durchzuführen ist.

Da man nicht alle Bearbeitungsvorschriften durch eindeutige Folgen von Anweisungen als Algorithmus darstellen kann, wurde im Laufe der Zeit der Begriff ausgedehnt. Man unterscheidet heute zwischen **deterministischen** und **nicht-deterministischen** Algorithmen. Enthält ein Algorithmus elementare Anweisungen, deren Ergebnis durch einen Zufallsmechanismus beeinflusst wird, so heißt dieser Algorithmus **nicht-deterministisch**. Liefert er bei der gleichen Eingabe immer die gleiche Ausgabe, so heißt er **deterministisch**.

Aus dieser Definition sind folgende **Eigenschaften**, die eine Vorschrift erfüllen muss um ein Algorithmus zu sein, ableitbar:

1. Vollständigkeit der Beschreibung

   Es muss eine komplette Anweisungsfolge vorliegen. Eine Bezugnahme auf unbekannte Information darf nicht enthalten sein.

2. Wiederholbarkeit des Algorithmus

   Im Sinne eines physikalischen Experiments muss man die Ausführungen eines Algorithmus beliebig oft wiederholen können und jede Wiederholung muss bei gleichen Eingabedaten das gleiche Resultat liefern (Reproduzierbarkeit). Ist ein Algorithmus endlich und definit so ergibt sich diese Forderung automatisch.

3. Korrektheit des Algorithmus

   Diese Forderung ist zwar selbstverständlich, aber in der Praxis ist die Korrektheit nur sehr schwer nachzuweisen. Man bedient sich daher Tests, bei denen für die vorgegebenen Eingabedaten das Ergebnis bereits bekannt ist, und vergleicht dann die erzeugten Ausgabedaten mit den Ergebnissen. (Ein solcher Test ist insofern problematisch, da alle möglichen Fälle abgedeckt werden müssen. Im Extremfall muss dieser Test sogar für jede mögliche Eingabe durchgeführt werden.) Der Begriff der Korrektheit nimmt zwar in der Literatur einen sehr großen Raum ein, da es nicht trivial ist, die Korrektheit nachzuweisen, er ist jedoch für die Definition eines Algorithmusbegriffes nicht erforderlich.

Kriterien für die **Güte** eines Algorithmus sind:

1. Die Effizienz der Beschreibung und der Ausführung (umständlich oder einfach).
2. Die Art der Ausführung einzelner Anweisungen (sequentiell oder parallel).
3. Die Komplexität bei der Ausführung. (Sie ist ein Maß für den Aufwand bei der Durchführung eines Algorithmus).

Darstellung von Algorithmen

1. Verbale Beschreibung

   Der Algorithmus wird in natürlicher Sprache beschrieben. Man kann dies in einer Aufzählung der Befehle machen oder zur Vereinfachung und um die Beschreibung eindeutig zu gestalten, die sprachliche Beschreibung formalisieren.

2. Programmablaufplan (PAP)

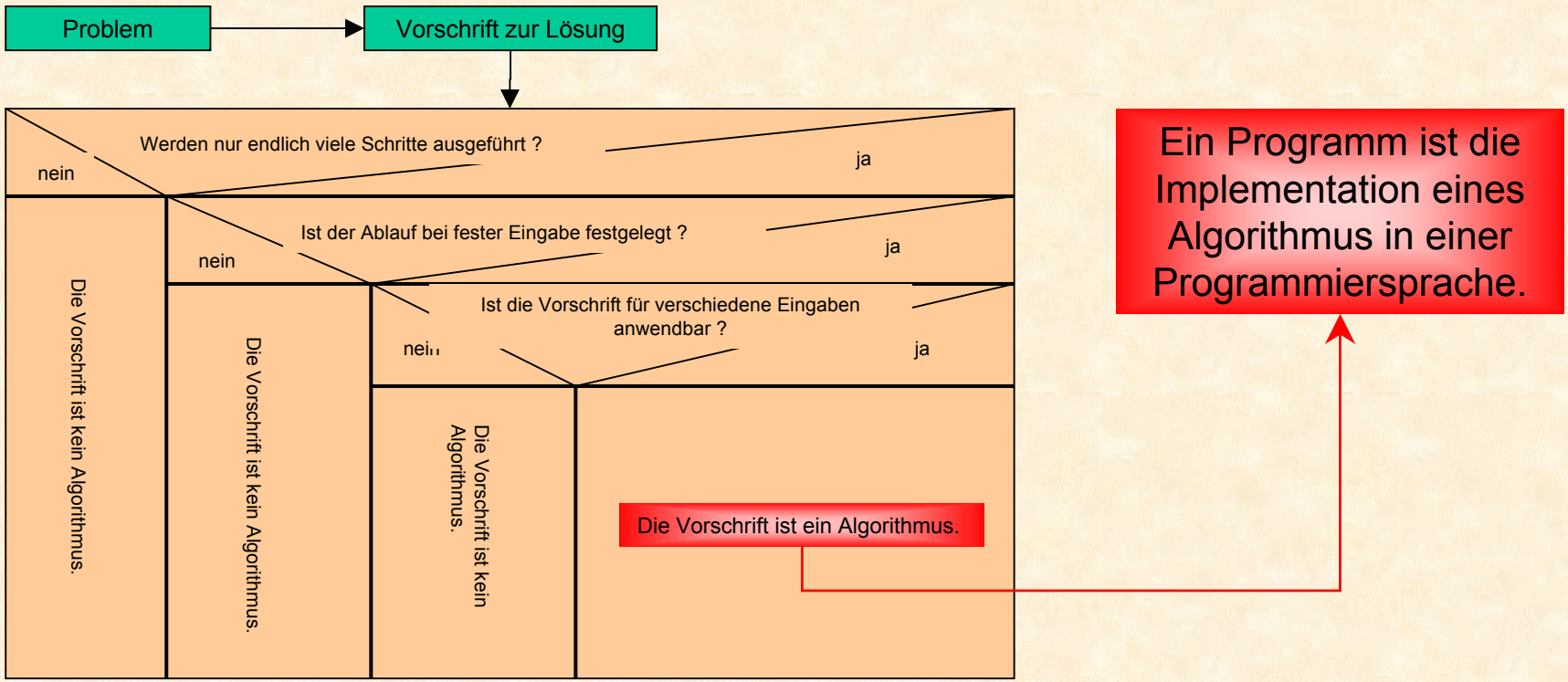
   Dies ist eine zeichnerische Darstellung mit genormten Symbolen (DIN 66001). Die Verarbeitungssymbole werden von geometrischen Formen umschlossen, deren Bedeutung genormt ist. Ihre Verknüpfung erfolgt durch eine Verbindungslinie, deren Durchlaufrichtung mit Pfeilen markiert ist.

3. Struktogramme

   Auch dies ist eine zeichnerische Darstellung (Nassi-Shneidermann), die jedoch auf Ablauflinien und Übergangsstellen verzichtet (bessere Übersichtlichkeit). Die Beschreibung des Algorithmus besteht aus geschachtelten Strukturelementen.

4. Programmiersprache

Ist eine Vorschrift ein Algorithmus ?



Eine **Programmiersprache** ist eine künstliche, normierte Sprache, die der Beschreibung von Algorithmen dient.
Durch ihre Anpassung an die menschliche Denkweise wird bei den Programmiersprachen das Schreiben von Algorithmen erleichtert; andererseits lässt sich aus einem Programm eine Beschreibung in natürlicher Sprache zurückgewinnen.
Die Normierung erfolgt durch die Festlegung der Syntax (Grammatik) und einer Semantik (Sinngehalt).

Maschinensprache

- Bestehen aus einem Satz durchnummerierter Befehle
- Programmierer muss genaueste Kenntnis über die Hardware (z.B. Speicheradressen) haben
- Befehlssatz ist abhängig von der verwendeten Hardware
- es gibt Befehle für
  - arithmetische Operationen
  - Vergleiche
  - Sprünge im Programm
  - Belegung von Speicheradressen
- **Programme werden vom Computer direkt „verstanden“**

maschinennahe Sprachen

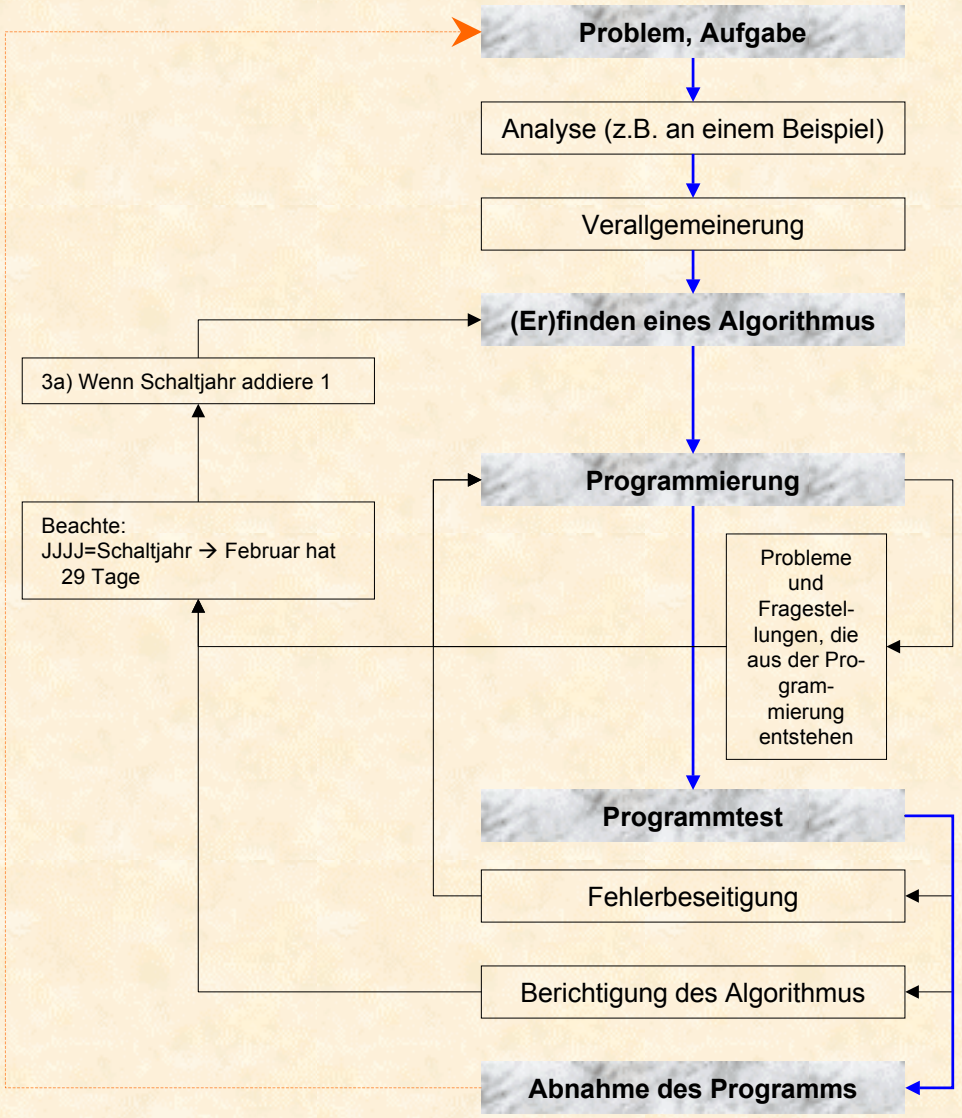
- Für die Maschinenbefehle gibt es eine Mnemonic (Klartextbefehl)
  - **ADD** R2,R3 addiert den Inhalt der Speicheradresse R2 zum Inhalt der Speicheradresse R3
- Programmierer muss genaueste Kenntnis über die Hardware (z.B. Speicheradressen) haben
- **Programme müssen für den Computer übersetzt werden (Assembler)**

höhere (algorithmische, problemorientierte) Sprachen

- Syntax und Semantik sind dem menschlichen Intellekt und der Umgangssprache angepasst
- innerhalb der Syntax sind Wortsymbole, Symbolzeichen, Standardbezeichnungen und frei wählbare Bezeichnungen möglich
- die Semantik der Symbole und Standardbezeichnungen ist vordefiniert
- es gibt Sprachen für bestimmte Anwendungsgebiete:
  - **FOR**mula **TRAN**slator - wiss.-techn. Anwendungen
  - **ALGO**rithmic **L**anguage - math. Anwendungen
  - **B**eginners **A**ll-purpose **S**ymbolic **I**nstruction **C**ode
  - **PASCAL**
  - **C**, **C++** (UNIX)
- **Programme müssen für den Computer übersetzt werden (Compiler, Interpreter)**

- Programmierung ist einfach aber sehr umständlich
- Fehlersuche ist sehr schwierig bis fast unmöglich
- neue Hardware = neues Programm
- es musste trickreich programmiert werden, um die Hardware weitestgehend ausnutzen (z.B. Speicher)
- Algorithmen können nur als unstrukturierte, lineare, monotone Sequenz von Befehlen dargestellt werden

Wie entsteht ein Programm ?



Zu einem Datum soll angegeben werden, um den wievielten Tag des Jahres es sich handelt.

17.5.2002 → 31+28+31+30+17=137

Addiere die Monatslängen bis einen Monat vor dem gegebenen Monat und addiere das Datum

1. Eingabe des Datum in der Form TT.MM.JJJJ
2. Addieren der Monatslängen bis MM-1
3. Addieren von TT
4. Ausgabe

...

- Das Datum als String → Zerlegung in TT und MM und JJJJ
- Sollen die Monatslängen durch 12 wenn-dann-Abfragen ermittelt werden ?
- Sollen die Monatslängen auf ein Array gelegt werden ?
- Gibt es eine Formel ?

Kriterien für die Güte eines Programms:

Zuverlässigkeit und Korrektheit

- korrekte Eingabe → korrektes Ergebnis
- Falsche Eingaben → werden erkannt, abgefangen und gemeldet, soweit möglich.
- Ausfälle der Hardware → können abgefangen oder in ihren Auswirkungen stark gemildert werden (Datensicherung, Wiederaufsetzen von Aufträgen).

lässt durch Modularisierung und Strukturierung und Beachtung des EVA-Prinzips erreichen.

Benutzerfreundlichkeit

- Für den gegebenen Benutzerkreis stellt das Programm keine zu hohen Anforderungen von der Ein- und Ausgabe her
- die Fragen sind dem Benutzer verständlich
- unvollständige Eingaben resultieren in Nachforderungen
- Fehlermeldungen sind im Klartext und geben Hinweise auf die Behebung des Fehlers (also nicht "ERROR EXIT 05 AT 5476777", sondern "FEHLER IN UNTERPROGRAMM NULLSTELLE WEGEN DIVISION DURCH 0")

Programme werden für Nutzer geschrieben und nicht für Programmierer.

Flexibilität

- neuer Benutzerwunsch → Änderungsarbeit sollte sich auf nur wenige Teilprobleme beschränken (Anpassbarkeit, Adaptibilität).
- Hardwarewechsel muss möglich sein (Übertragbarkeit, Portabilität)
- Betriebssystemwechsel sollte möglich sein (Win98 → Win 2000 → Win XP ; Win xx → Linux)

lässt durch Modularisierung und Strukturierung leichter erreichen.

Lesbarkeit

- aussagekräftige Variablenbezeichner verwenden
- relativ kurze Module schreiben
- viele Kommentare (Remarks) im Quelltext verwenden
- Programmteile sauber strukturieren (z.B. Einrücken)

Lesbarkeit ist Voraussetzung für Wartbarkeit.

Effizienz

Das Programm kann an der benötigten Laufzeit und an den Speicheranforderungen gemessen werden. Auf dieses Kriterium darf aber erst dann geachtet werden, wenn eine fertige Version des Programms vorliegt, die allen anderen Anforderungen genügt. Erst dann können zusätzliche Effizienzverbesserungen, ausführlich erläutert, toleriert werden.