Informatik 1

Programmiersprachen

Jonas Miederer

DHBW Stuttgart - Campus Horb

26. November 2020

DHBW
Duale Hochschule
Baden-Württemberg

Outline:

DHBW
Duale Hochschule
Baden-Württemberg

Inhaltsverzeichnis

DHBW
Duale Hochschule
Baden-Württemberg

Programmiersprachen

Mithilfe von Computerprogrammen können komplexe wohldefinierte Problemstellungen (teil)automatisiert gelöst werden. Voraussetzung dafür ist, dass

- das Problem konkret beschrieben werden kann
- sich das Problem ihm Rahmen der Möglichkeiten einer Programmiersprache abbilden lässt
- es mindestens einen Lösungsweg gibt
- der Lösungsweg endlich ist

DHBW
Duale Hochschule
Baden-Württemberg

Einordnung von Programmiersprachen

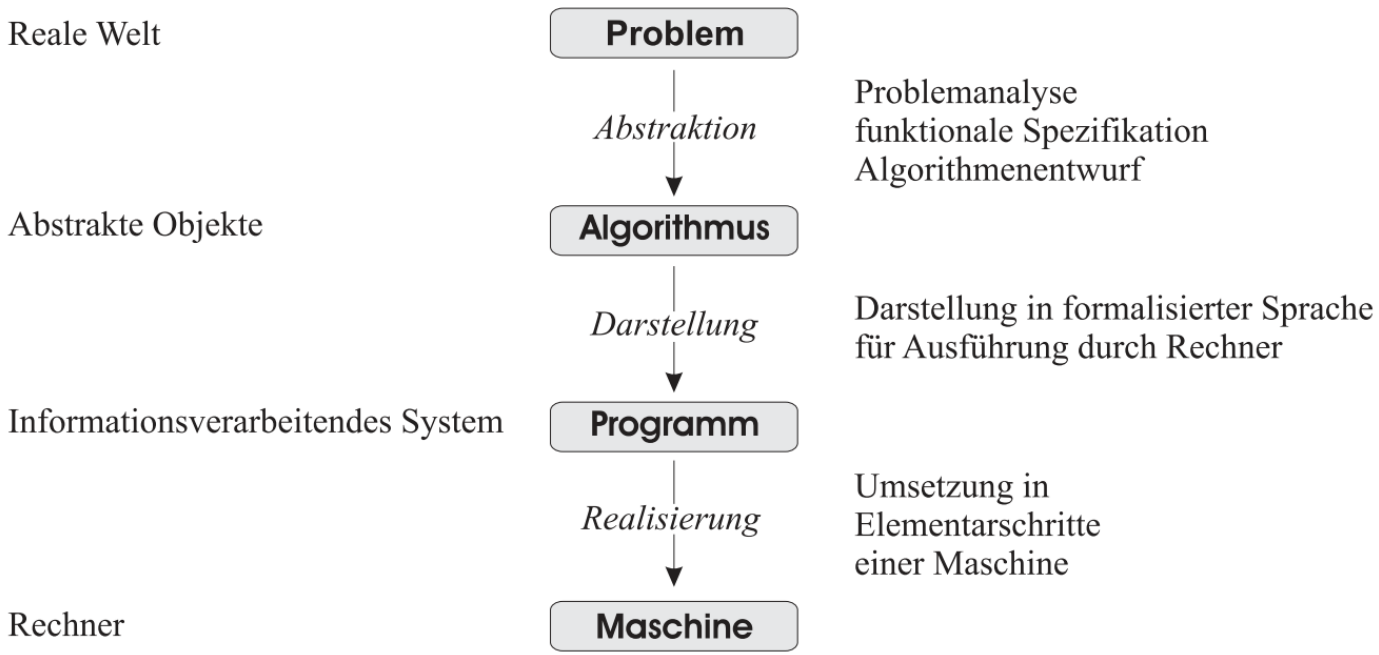

Abbildung 1: Vorgehensweise zur Lösung von Problemen in der Informatik [1]

Einordnung von Programmiersprachen

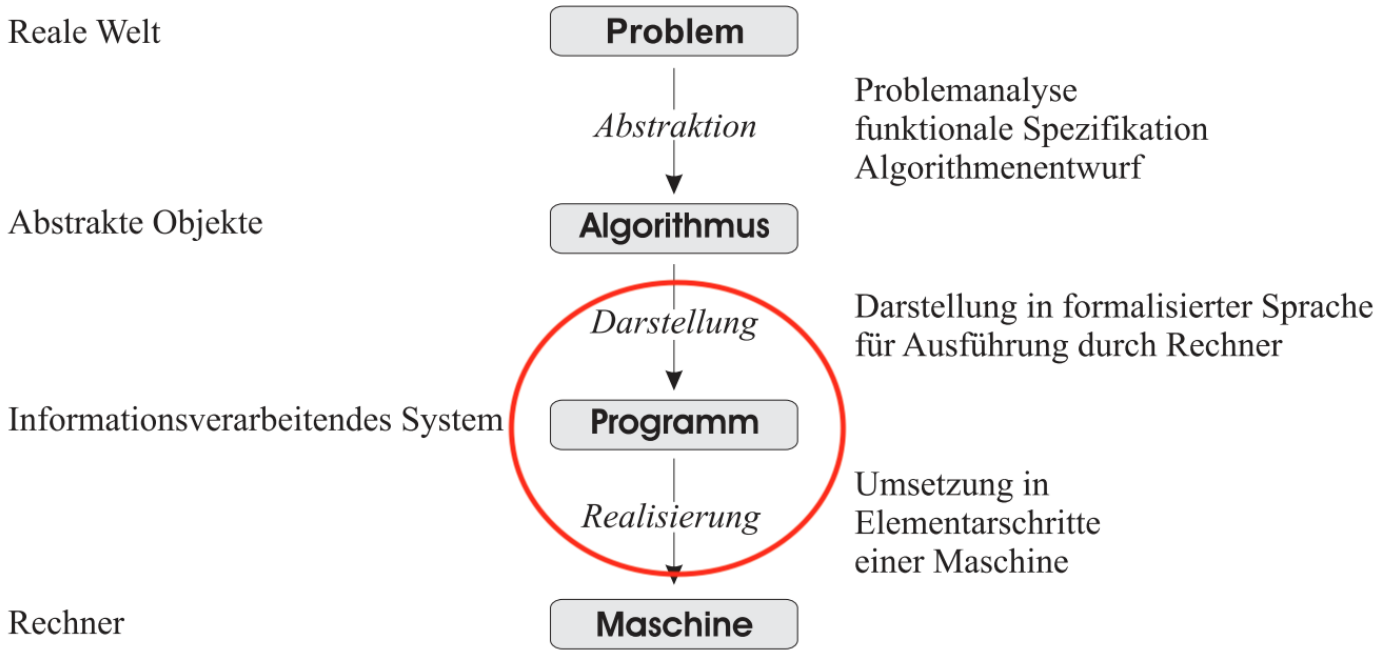

Abbildung 2: Vorgehensweise zur Lösung von Problemen in der Informatik [1] (bearbeitet)

Zweck von Programmiersprachen

Programmiersprachen helfen also dabei, das Problem, welches zunächst abstrakt beschrieben wird, formal darzustellen und in eine Form zu übertragen, die für das Computersystem verarbeitbar ist.

Einordnung von Programmiersprachen

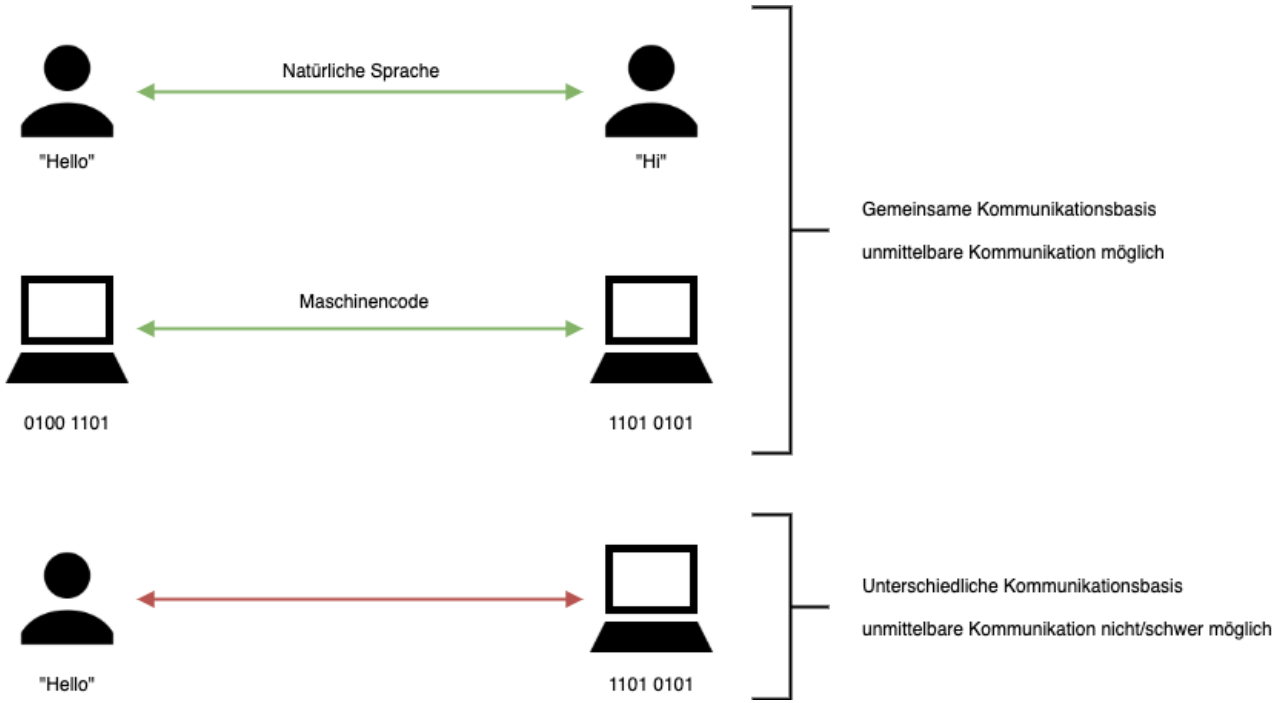

Abbildung 3: Kommunikationswege zwischen Mensch und Computer (I)

Einordnung von Programmiersprachen

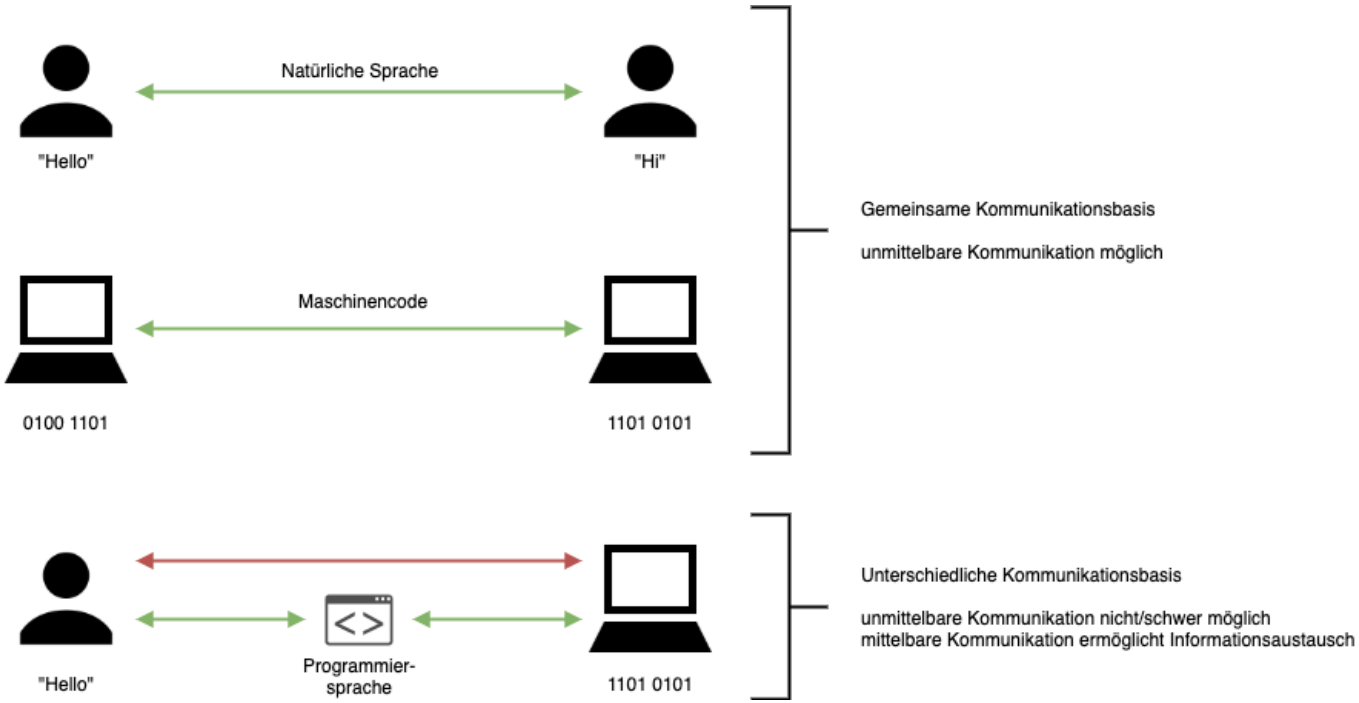

Abbildung 4: Kommunikationswege zwischen Mensch und Computer (II)

Inhaltsverzeichnis

DHBW
Duale Hochschule
Baden-Württemberg

Programmiersprachen

Abbildung 5: Word-Cloud zur Vielfalt an Programmiersprachen [2]

Beliebtheit von Programmiersprachen

Dec 2019	Dec 2018	Change	Programming Language	Ratings	Change
1	1		Java	17.253%	+1.32%
2	2		C	16.086%	+1.80%
3	3		Python	10.308%	+1.93%
4	4		C++	6.196%	-1.37%
5	6	^	C#	4.801%	+1.35%
6	5	v	Visual Basic .NET	4.743%	-2.38%
7	7		JavaScript	2.090%	-0.97%
8	8		PHP	2.048%	-0.39%
9	9		SQL	1.843%	-0.34%
10	14	^^	Swift	1.490%	+0.27%

Abbildung 6: TIOBE-Index zur Beliebtheit von Programmiersprachen 12/2019 [3]

Hinweis

Der TIOBE-Index stellt keine wissenschaftliche Auswertung zur Beliebtheit von Programmiersprachen dar, die Ergebnisse sind lediglich auf Auswertungen von Suchanfragen gestützt. Dennoch bietet der Index einen groben Überblick.

Beliebtheit von Programmiersprachen

Nov 2020	Nov 2019	Change	Programming Language	Ratings	Change
1	2	↑	C	16.21%	+0.17%
2	3	↑	Python	12.12%	+2.27%
3	1	↓	Java	11.68%	-4.57%
4	4		C++	7.60%	+1.99%
5	5		C#	4.67%	+0.36%
6	6		Visual Basic	4.01%	-0.22%
7	7		JavaScript	2.03%	+0.10%
8	8		PHP	1.79%	+0.07%
9	16	⇈	R	1.64%	+0.66%
10	9	↓	SQL	1.54%	-0.15%

Abbildung 7: TIOBE-Index zur Beliebtheit von Programmiersprachen 11/2020 [3]

Beliebtheit von Programmiersprachen

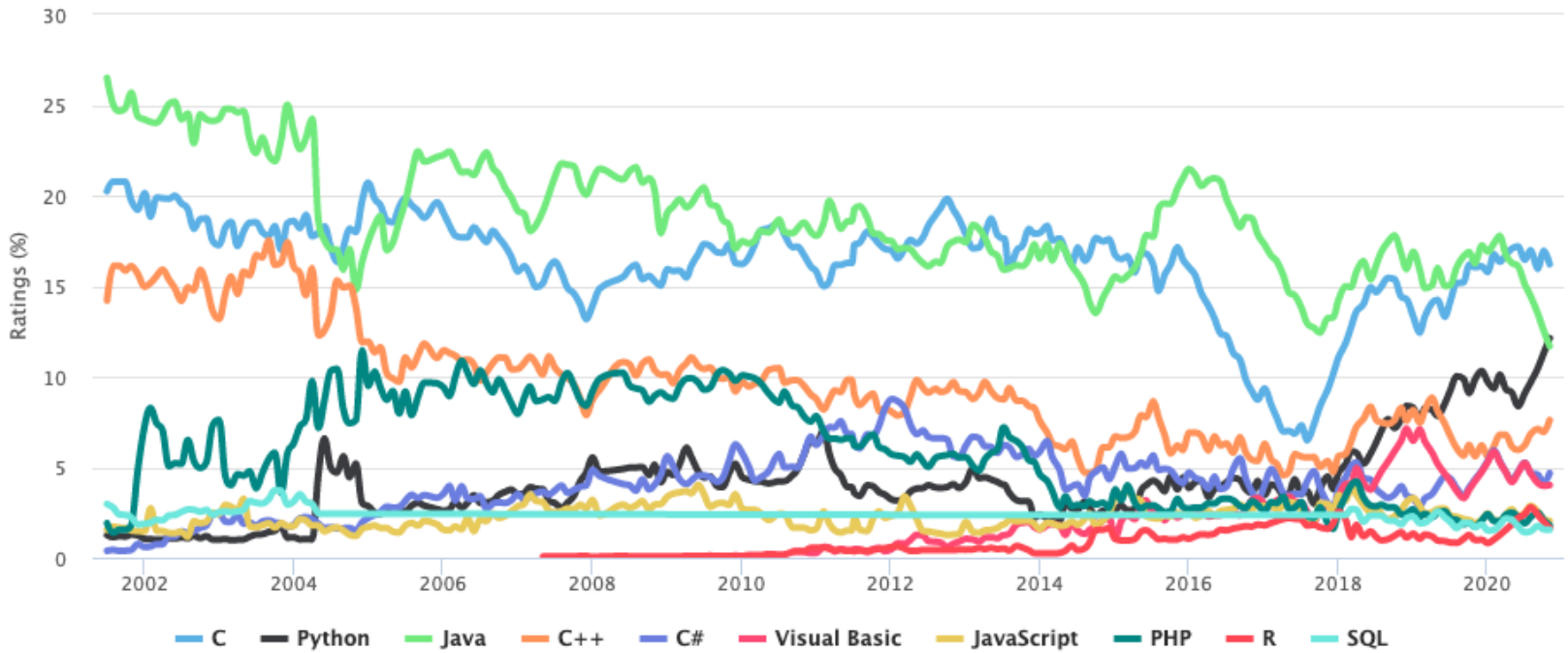

Abbildung 8: TIOBE-Index zur Beliebtheit von Programmiersprachen Juli 2001 - November 2020 [4]

Vielfältigkeit und Anzahl von Programmiersprachen

Why NASA Needs a Programmer Fluent In 60-Year-Old Languages

To keep the Voyager 1 and 2 crafts going, NASA's new hire has to know FORTRAN and assembly languages.

By John Wenz Oct 29, 2015

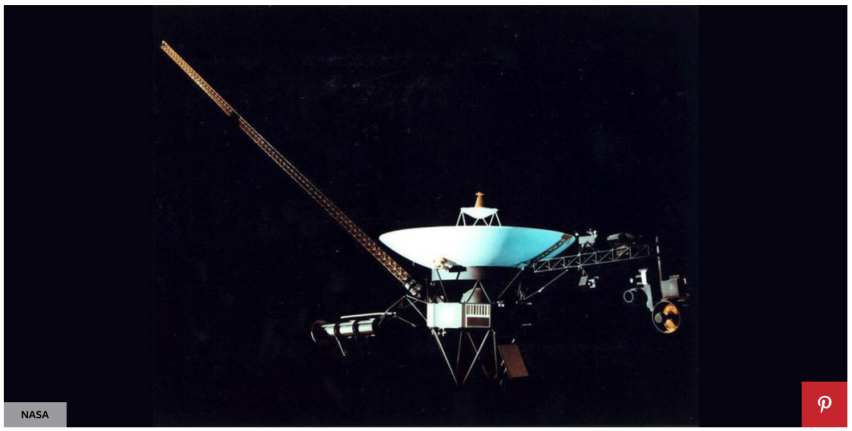

Abbildung 9: Bericht über die Suche der NASA nach einem COBOL-Entwickler [5]

Frage 1: Was hat das mit Programmiersprachen zu tun?

Romeo, a young man with a remarkable patience.
Juliet , a likewise young woman of remarkable grace.
Ophelia, a remarkable woman much in dispute with Hamlet.
Hamlet, the flatterer of Andersen Insulting A/S.

Act I: Hamlet's insults and flattery .

Scene I: The insulting of Romeo.

[Enter Hamlet and Romeo]

Hamlet:
You lying stupid fatherless big smelly half-witted coward!
You are as stupid as the difference between a handsome rich brave hero and thyself ! Speak your mind!

You are as brave as the sum of your fat little stuffed misused dusty old rotten codpiece and a beautiful fair warm peaceful sunny summer's day. You are as healthy as the difference between the sum of the sweetest reddest rose and my father and yourself ! Speak your mind!

You are as cowardly as the sum of yourself and the difference between a big mighty proud kingdom and a horse. Speak your mind.

Speak your mind!

[Exit Romeo]

[Exeunt]

Frage 2: Was hat das mit Programmiersprachen zu tun?

What are we in this country
Hillary speaks nothing but lies
But look at me I came to this election to make guys
believe again
believe in fact
if all of us real lies the light ; : say "VOTE TRUMP"!
but I know we should be free
else the result will be bad: all the work of George
Washington was for nothing
so this election say "Hello, World" say "TRUMP FOR PRESIDENT"!
America is great.

Frage 3: Was hat das mit Programmiersprachen zu tun?

Abbildung 10: [6]

Frage 4: Was hat das mit Programmiersprachen zu tun?

```
+[-[<<[+[---->]-[<<<]]]>>>-]>-.---.>..>.<<<<<-.<+.>>>>>>.>.<<.<-.
```

Inhaltsverzeichnis

DHBW
Duale Hochschule
Baden-Württemberg

Eigenschaften

Alle Programmiersprachen müssen verschiedene Voraussetzungen erfüllen, damit es sich tatsächlich um eine Programmiersprache handelt.

Hinweis

Die im Folgenden aufgeführten Eigenschaften stellen lediglich die "Mindestvoraussetzungen" von Programmiersprachen dar. Die Programmiersprachen besitzen darüber hinaus noch viele weitere Eigenschaften, die sich jedoch unterscheiden können bzw. nicht zwingend in jeder Programmiersprache vorhanden sein müssen.

Eingabe & Ausgabe

Jede Programmiersprache muss **Eingaben** verarbeiten können (Informationsquelle). Dabei spielt es keine Rolle, ob die Eingabe durch eine manuelle Nutzereingabe (z.B. Text durch Tastatur/Maus, Spracheingabe, Dokumentenscan, ...) oder durch automatisiertes Einlesen von Daten (z.B. Datenbank, Datei, Programmierschnittstelle, ...) erfolgt.

Ebenso muss die **Ausgabe** von Informationen möglich sein, um dem Nutzer oder einem anderen System die Ergebnisse zur Verfügung zu stellen (Informationssenke). Auch hierfür gibt es genauso wie bei der Eingabe verschiedene Möglichkeiten (z.B. Ausgabe am Bildschirm, Audioausgabe, Datenbank, ...)

Auf welche Art und Weise und in welchem Format die Ein- und Ausgabe erfolgt ist jeweils abhängig vom Anwendungsfall und der Umsetzung im Programm und ist daher nicht vorgegeben.

DHBW
Duale Hochschule
Baden-Württemberg

Deklaration von Variablen

Um Informationen innerhalb eines Programms verarbeiten zu können, müssen die Daten während der Verarbeitung zwischengespeichert werden. Hierzu werden Variablen genutzt, um Informationen referenzieren (darauf zugreifen) zu können.

Variable

Speichereinheit zur Aufnahme, Zwischenspeicherung und Modifikation von Daten(werten)

Deklaration

Zuweisung / Zuordnung eines bestimmten Werts zu einer bestimmten Variablen (zu einem bestimmten Variablennamen)

DHBW
Duale Hochschule
Baden-Württemberg

Mathematische Grundoperationen

Wenn Informationen nicht nur (zwischen)gespeichert, sondern auch verarbeitet werden sollen, sind mathematische Grundoperationen unabdingbar. Auf Maschinencode-Ebene werden beinahe sämtliche Operationen durch Addition ausgeführt. Aber auch in höheren Programmiersprachen sind keine Berechnungen ohne mathematische Standardfunktionen möglich.

Zeichenkettenverarbeitung

Während mathematische Grundoperationen zur Verarbeitung von Zahlen benötigt werden, müssen auch Funktionen zur Verarbeitung und Speicherung von Zeichenketten (Strings) bereitstehen.

String

Ein String (Zeichenkette) repräsentiert in der Programmierung Text. Dazu können Buchstaben, Sonderzeichen, Satzzeichen und Steuerzeichen gehören

Steueranweisungen

Zu den Steueranweisungen gehören alle Befehle und Operationen, die den Ablauf eines Programms beschreiben und beeinflussen.

Dazu können etwa Schleifen, bedingte Anweisungen, Funktionen / Prozeduren, und andere Sprachkonstrukte gehören.

Zusammenfassung

Zu den Eigenschaften jeder Programmiersprache gehören folgende Grundfunktionalitäten:

- Ein- und Ausgabe von Informationen
- Deklaration von Variablen und Verarbeitung deren Werten
- Mathematische Grundoperationen zur Verarbeitung von Zahlenwerten
- Zeichenkettenverarbeitung zur Verarbeitung von Strings
- Steueranweisungen zur Programmablaufbeschreibung

DHBW
Duale Hochschule
Baden-Württemberg

Inhaltsverzeichnis

DHBW
Duale Hochschule
Baden-Württemberg

Einordnung von Programmiersprachen

Programmiersprachen können aufgrund unterschiedlicher Konzepte, Eigenschaften und Merkmale unterschieden und so in diverse Gruppen eingeteilt werden. In manchen Bereichen ist eine klare oder eindeutige Zuordnung einer bestimmten Programmiersprache zu einer Gruppe nicht oder nur schwer möglich.

Im Folgenden werden die wichtigsten Merkmale, anhand deren die Programmiersprachen unterschieden werden können, dargestellt. Die Aufzählung hat nicht den Anspruch der Vollständigkeit, da die Unterschiede auch subjektiv erhoben werden können.

Programmierparadigmen

Programmiersprachen können nach unterschiedlichen Paradigmen bzw. Mustern unterteilt werden. Ein Programmierparadigma beschreibt dabei einen bestimmten Stil, an dem sich die Programmierung, Umsetzung der Lösung und Implementierung des Codes orientiert.

Meist werden unterschiedliche Programmiersprachen einem Programmierparadigma zugeordnet, in manchen Fällen ist das Paradigma von der Sprache aber nicht fest vorgegeben, sodass dem Entwickler die Entscheidung für ein Paradigma überlassen ist.

Programmiersprachen können auch nach mehreren Programmierparadigmen gestaltet sein.

DHBW
Duale Hochschule
Baden-Württemberg

Programmierparadigmen

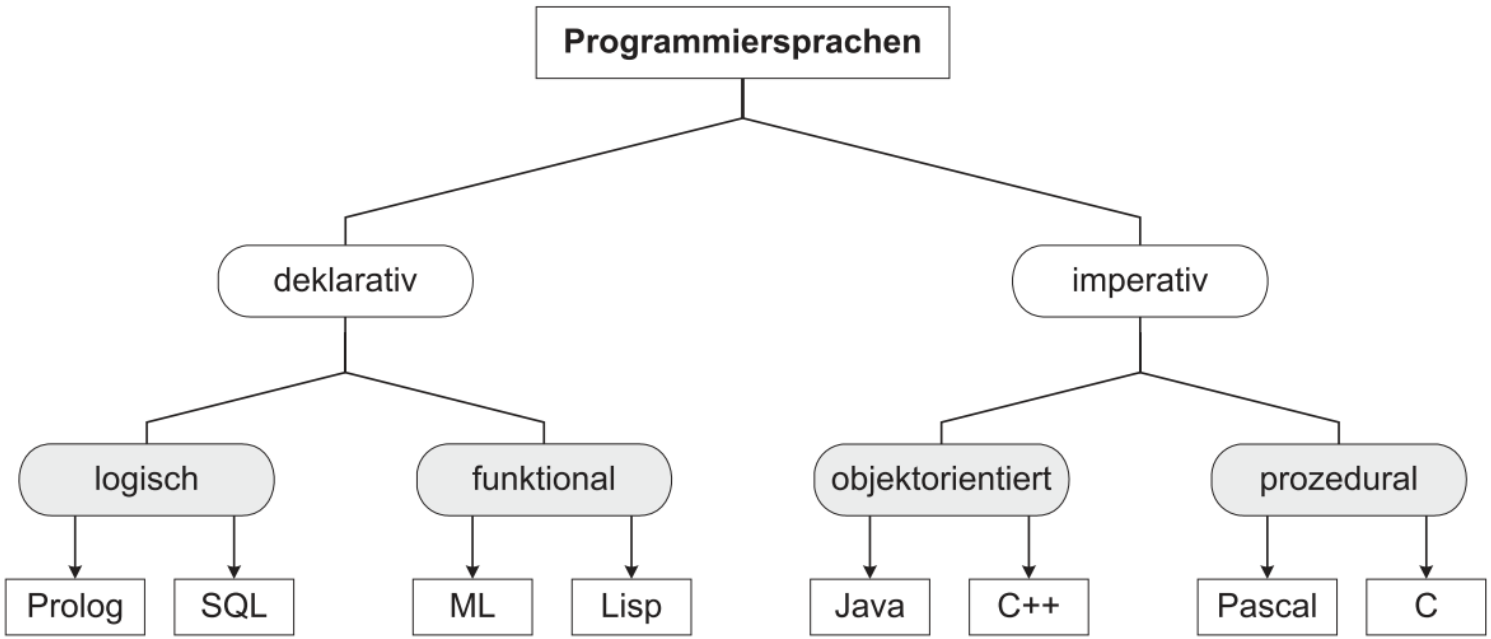

Abbildung 11: Gliederung der Programmiersprachen nach unterschiedlichen Paradigmen [1]

Programmierparadigmen: Imperativ (I)

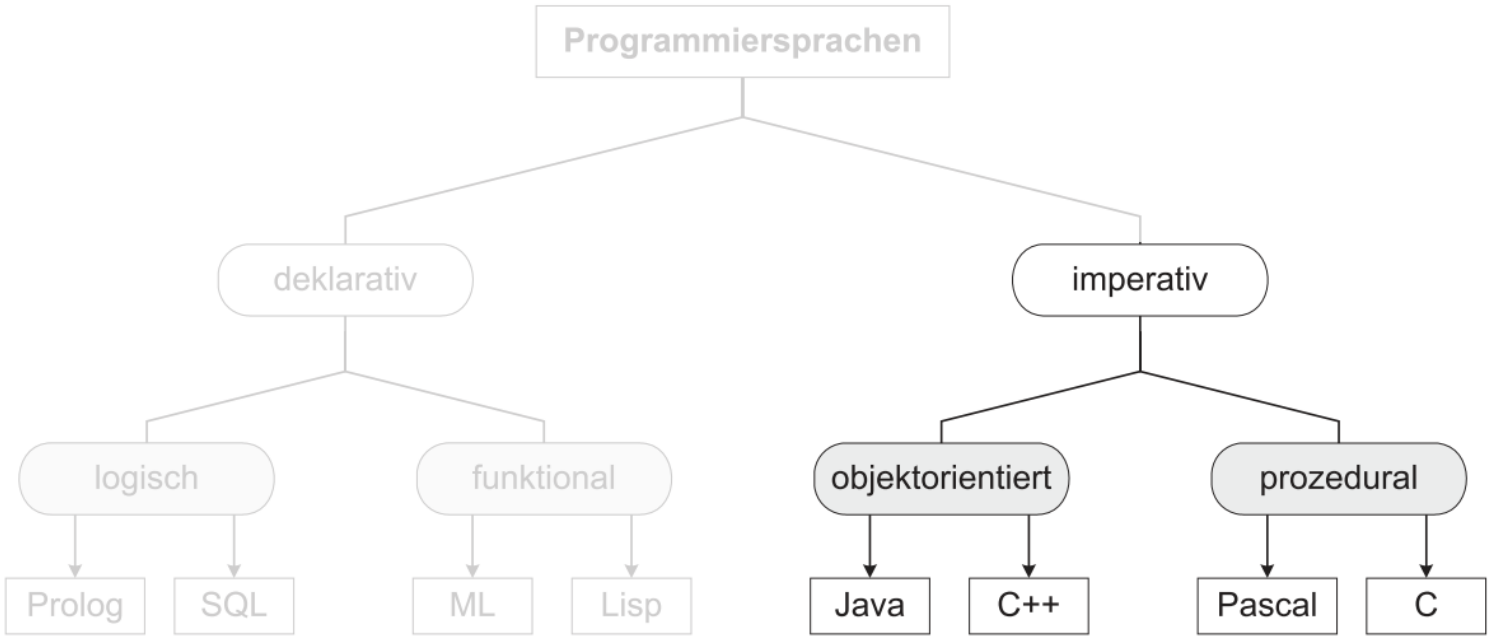

Abbildung 12: Gliederung der Programmiersprachen nach unterschiedlichen Paradigmen [1] (bearbeitet)

Programmierparadigmen: Imperativ (II)

Bei der *imperativen Programmierung* beschreibt der Programmierer, was der Reihe nach ausgeführt werden soll. Er ist also sowohl für die zugrundeliegende Funktionalität als auch für die ordnungsgemäße Reihenfolge dieser Anweisungen zuständig.

Die imperative Programmierung stellt das älteste und weitverbreitetste Paradigma dar, da es große Ähnlichkeiten zum menschl. Denkprozess aufweist.

Imperative Programmierung

Ursprung: *imperare* (lat.): *'anordnen', 'befehlen'*
Der Programmierer beschreibt, **wie** eine Lösung zu einem bestimmten Problem ermittelt wird.

DHBW Duale Hochschule Baden-Württemberg

Programmierparadigmen: Imperativ (III)

Häufig wird innerhalb der imperativen Programmierung zwischen zwei weiteren (konträren) Paradigmen unterschieden:

Prozedural

- Programmablauf kann in Teilprobleme zerlegt werden
- Wichtig zur Übersichtlichkeit, Wartbarkeit und Vermeidung von Redundanz
- Daten und Funktionen haben keinen definierten Zusammenhalt

Objektorientiert

- Funktionalitäten werden innerhalb von Klassen nach Zuständigkeit getrennt
- Über einfache Datenstrukturen hinausgehende Abstraktion der Daten und Funktionalität
- Daten und Funktionen werden in Klassen / Objekten zusammengefasst

DHBW Duale Hochschule Baden-Württemberg

Programmierparadigmen: Deklarativ (I)

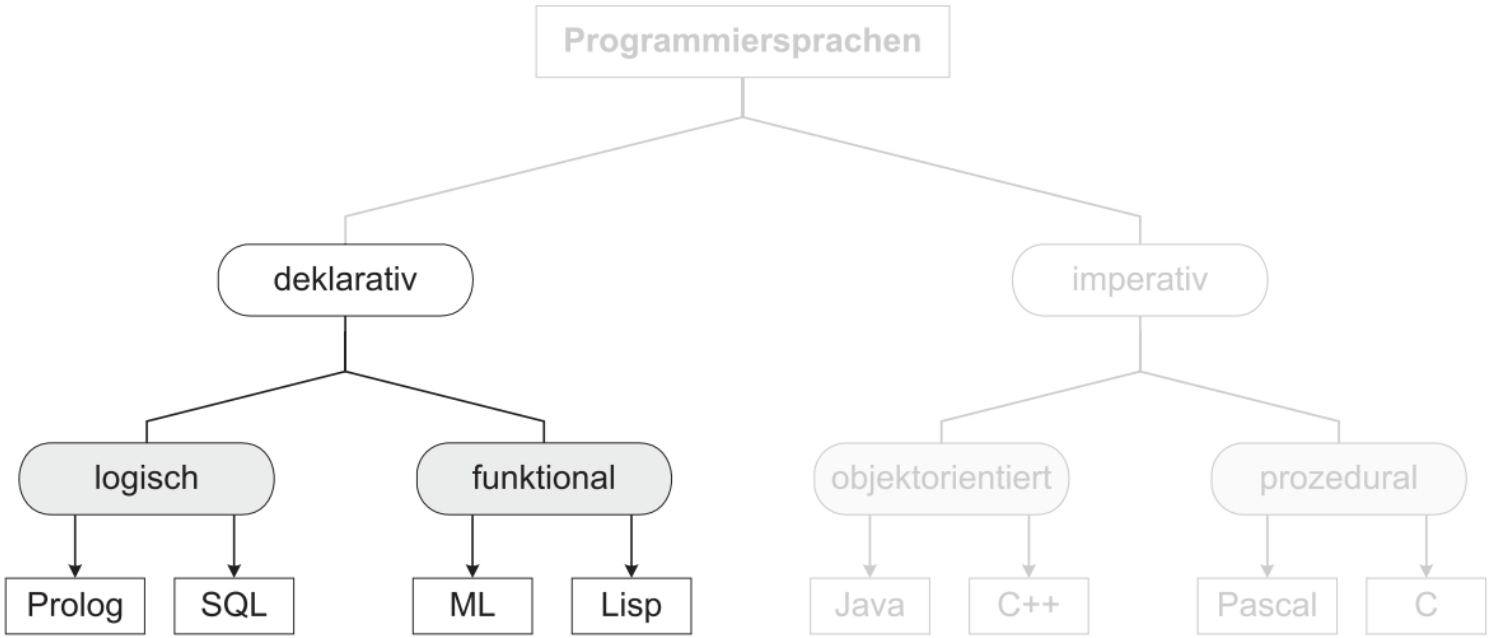

Abbildung 13: Gliederung der Programmiersprachen nach unterschiedlichen Paradigmen [1] (bearbeitet)

Programmierparadigmen: Deklarativ (II)

Bei der *deklarativen Programmierung* beschreibt der Programmierer nicht, was der Reihe nach ausgeführt werden soll, sondern *was* berechnet werden soll. Dadurch steht hier vielmehr die Beschreibung des Problems als die explizite Beschreibung des Lösungswegs im Vordergrund.

Im Gegensatz zur imperativen Programmierung lassen sich hier Arbeits- und Steuermechanismen trennen, sodass dieses Paradigma rechnerunabhängig ist.

Deklarative Programmierung

Ursprung: *declarar* (lat.): *'erklären'*
Der Programmierer beschreibt, **was** berechnet werden soll und nicht, wie der entsprechende Rechenweg dafür aussieht.

Programmierparadigmen: Imperativ vs. Deklarativ

Beispiel: Durchsuche Bibliothekskatalog nach Büchern mit dem Titel "Programmierung"
[1]

Beispiel: Imperative Programmierung

1. Nimm Buch
2. Prüfe, ob Titel = "Programmierung"
3. Falls JA, notiere Autor
4. Prüfe, ob letztes Buch
5. Falls NEIN, gehe zu (1)
6. Falls JA →Ende

Beispiel: Deklarative Programmierung

1. Suche alle Bücher, für die gilt (Titel = "Programmierung")

DHBW Duale Hochschule Baden-Württemberg

[1] angelehnt an [7]

Maschinennähe

Beschreibt den Abstraktionsgrad bzw. die Nähe der Programmiersprache zum darunterliegenden System

Maschinennahe Sprachen ("Low-Level Sprachen") bieten mehr Kontrolle und können tiefer in Systemfunktionalitäten eingreifen, sind aber meist auch komplexer und schwerer zu beherrschen als höhere Sprachen ("High-Level Sprachen")

Low-Level Sprachen

- Geringer oder kein Abstraktionsgrad
- Direkte Speicherzugriffe, Kernelzugriffe, ... möglich
- Mehr Kontrolle, höhere Komplexität
- Beispiele: Maschinencode, Algol, Assembler, C

High-Level Sprachen

- Hoher Abstraktionsgrad
- Keine direkten Speicherzugriffe, Kernelzugriffe, ... möglich
- Weniger Kontrolle, geringere Komplexität
- Beispiele: Java, Python, C++

DHBW Duale Hochschule Baden-Württemberg

Computersprachen

Nicht alle Sprachen, die häufig als "Programmiersprachen" bezeichnet werden, sind auch tatsächlich Programmiersprachen, da sie beispielsweise nicht alle Eigenschaften einer Programmiersprache erfüllen (siehe Abs. 3). Daher können Sprachen auch entsprechend ihres Zwecks untergliedert werden:

- **Programmiersprache**: Kann komplexe Programmabläufe darstellen, besitzt alle beschriebenen Eigenschaften einer Programmiersprache (Beispiel: Python, Java, C)
- **Datenbanksprache**: Wird zur Kommunikation mit einem Datenbanksystem verwendet (Beispiel: SQL)
- **Auszeichnungssprache**: Beschreibt den Aufbau und die Struktur von Dokumenten (Beispiel: HTML, XML)
- **Stylesheet-Sprache**: Beschreibt das Erscheinungsbild von Dokumenten unabhängig vom Inhalt (Beispiel: CSS)

DHBW
Duale Hochschule
Baden-Württemberg

Typisierung / Typsysteme

In jeder Programmiersprache können Variablen erzeugt werden, denen unterschiedliche Arten von Werten (**Datentypen**) zugewiesen werden können. Dazu zählen bspw. Zahlen (Ganzzahlen, Fließkommazahlen), Zeichenketten (Strings) und Wahrheitswerte. Die Typisierung einer Programmiersprache dient zur korrekten Verwendung dieser Datentypen.

Der Umgang mit den Datentypen kann sich zwischen den Programmiersprachen unterscheiden.

Typisierung / Typsysteme: Typenlose Sprachen

Manche Sprachen sind typenlose Sprachen. Derartige Programmiersprachen verfügen über keine differenzierten Datentypen. Das ist für den Entwickler einerseits sehr bequem, da er sich nicht um die entsprechenden Datentypen kümmern muss, führt aber auch schnell zu Fehlern, die aufgrund fehlender Typbezeichnungen manchmal nur schwer festzustellen und zu finden sind.

Beispiel: Assembler

Typisierung / Typsysteme: Typisierte Sprachen

Die meisten Programmiersprachen sind typisiert, d.h. alle Variablen besitzen jederzeit einen bestimmten Datentyp.

Typsysteme können weiter klassifiziert werden:

- Statische vs. dynamische Typisierung: Typprüfungen finden zur Übersetzungszeit (statisch) oder zur Laufzeit (dynamisch) statt
- Explizite vs. implizite Typisierung: Datentypen werden explizit genannt oder implizit ermittelt
- Starke vs. schwache Typisierung: Verlustbehaftete Typumwandlungen können nicht (stark) / können (schwach) durchgeführt werden

DHBW Duale Hochschule Baden-Württemberg

Anwendungsbereich

Manche Programmiersprachen werden nur domänenspezifisch eingesetzt, andere sind universell einsetzbar [2].

- Websprachen (Serverseitig): **PHP**, **Ruby**, **node.js**, Python, Java
- Websprachen (Clientseitig): **JavaScript**
- Mobile Applications: **Kotlin** / Java (Android), **Swift** / Objective-C (iOS)
- Business Applications: Java, C++, C#
- Datenbanken: **SQL**

[2] Fettgedruckte Sprachen sind explizit für den jeweiligen Einsatzbereich entwickelt und werden ausschließlich dort verwendet

Antwort 1: Was hat das mit Programmiersprachen zu tun?

```
Romeo, a young man with a remarkable patience.
Juliet, a likewise young woman of remarkable grace.
Ophelia, a remarkable woman much in dispute with Hamlet.
Hamlet, the flatterer of Andersen Insulting A/S.

                    Act I: Hamlet's insults and flattery.

                    Scene I: The insulting of Romeo.

[Enter Hamlet and Romeo]

Hamlet:
 You lying stupid fatherless big smelly half-witted coward!
 You are as stupid as the difference between a handsome rich brave
 hero and thyself! Speak your mind!

 You are as brave as the sum of your fat little stuffed misused dusty
 old rotten codpiece and a beautiful fair warm peaceful sunny summer's
 day. You are as healthy as the difference between the sum of the
 sweetest reddest rose and my father and yourself! Speak your mind!

 You are as cowardly as the sum of yourself and the difference
 between a big mighty proud kingdom and a horse. Speak your mind.

 Speak your mind!

[Exit Romeo]

[Exeunt]
```

The Shakespeare Programming Language

A programming language created with the design goal to make the source code resemble Shakespeare plays.

The characters in the play are variables. If you want to assign a character, let's say Hamlet, a negative value, you put him and another character on the stage and let that character insult Hamlet.

Input and output is done be having someone tell a character to listen their heart and speak their mind. The language contains conditionals, where characters ask each other questions, and jumps, where they decide to go to a specific act or scene. Characters are also stacks that can be pushed and popped.

Abbildung 14: The Shakespeare Programming Language [8]

Antwort 2: Was hat das mit Programmiersprachen zu tun?

```
What are we in this country
Hillary speaks nothing but lies
But look at me I came to this election to make guys
believe again
believe in fact
if all of us real lies the light; : say "VOTE TRUMP"!
but I know we should be free
else the result will be bad: all the work of George
Washington was for nothing
so this election say "Hello, World" say "TRUMP FOR PRESIDENT"!
America is great.
```

TrumpScript

Make Python great again

Mission

TrumpScript is a language based upon the illustrious Donald Trump. As the undeniably best US President, we found that the current field of programming languages does not include any that Trump's glorious golden combover would approve of.

TrumpScript is our solution to this. It's the programming language Trump would approve of. Just like he is going to make America great again, we hope our efforts will make programming great again.

Abbildung 15: TrumpScript [9]

Antwort 3: Was hat das mit Programmiersprachen zu tun?

Abbildung 16: [6]

Introduction

Piet is a programming language in which programs look like abstract paintings. The language is named after Piet Mondrian, who pioneered the field of geometric abstract art. I would have liked to call the language Mondrian, but someone beat me to it with a rather mundane-looking scripting language. Oh well, we can't all be esoteric language writers I suppose.

Abbildung 17: Piet [10]

Antwort 4: Was hat das mit Programmiersprachen zu tun?

```
+[-[<<[+[---->]-[<<<]]]>>>-]>-.---.>..>.<<<<-.<+.>>>>>.>.<<.<-.
```

Brainfuck

Brainfuck (dt: „Hirnfick“) ist eine esoterische Programmiersprache, die der Schweizer Urban Müller im Jahre 1993 entwarf. Die Sprache wird auch als Brainf*ck, Brainf*** oder BF bezeichnet.

Brainfuck ist für den produktiven Einsatz viel zu umständlich und zu ineffizient, aber geeignet, um die Methodik von Softwareentwicklung zu schulen. Speziell zeichnet sich Brainfuck durch ein extrem einfaches Sprachkonzept und hochkompakte Realisierung des Compilers aus, gleichzeitig wurde aber die (im berechenbarkeitstheoretischen Sinne) universelle Einsetzbarkeit nicht eingeschränkt.

Abbildung 18: Brainfuck [11]

Inhaltsverzeichnis

DHBW Duale Hochschule Baden-Württemberg

Bibliographie I

Heinrich Müller und Frank Weichert. *Vorkurs Informatik*. Wiesbaden: Springer Fachmedien Wiesbaden, 2015. ISBN: 978-3-658-08101-0. DOI: 10.1007/978-3-658-08102-7. URL: http://link.springer.com/10.1007/978-3-658-08102-7.

Programming languages. 2015. URL: https://upload.wikimedia.org/wikipedia/commons/0/0d/Prog-languages.png.

TIOBE Index for December 2019. Dez. 2019. URL: https://www.tiobe.com/tiobe-index/.

TIOBE Programming Community Index. Dez. 2019. URL: https://www.tiobe.com/tiobe-index/.

John Wenz. *Why NASA Needs a Programmer Fluent In 60-Year-Old Languages*. Okt. 2015. URL: https://www.popularmechanics.com/space/a17991/voyager-1-voyager-2-retiring-engineer/.

Thomas Schoch. *Programm in der Programmiersprache Piet, Ausgabe: „Hello, World!“*. Feb. 2006. URL: https://commons.wikimedia.org/wiki/File:Piet_Program_Hello_World(1).gif.

DHBW Duale Hochschule Baden-Württemberg

Bibliographie II

J.M. Leimeister. *Einführung in die Wirtschaftsinformatik*. Springer Berlin Heidelberg, 2015. ISBN: 9783540778479.

Karl Wiberg und Jon Åslund. *The Shakespeare Programming Language*. URL: http://shakespearelang.sourceforge.net/.

Lewis Cannon. *TrumpScript*. 2015. URL: https://github.com/samshadwell/TrumpScript.

David Morgan-Mar. *Piet*. 2018. URL: https://www.dangermouse.net/esoteric/piet.html.

Wikipedia. *Brainfuck*. 2019. URL: https://de.wikipedia.org/wiki/Brainfuck.

DHBW
Duale Hochschule
Baden-Württemberg