Geschichte der Programmiersprachen

- *Erster Vorläufer:* das Plankalkül, 1942 von K. Zuse, erst 1975 erstmals vollständig beschrieben und implementiert, keine praktische Relevanz

- *Sprachen der 1. Generation (1.GL):*
  Maschinensprachen: binär, prozessorspezifisch

  Operationen: Verschieben von Daten zwischen Speicher und Registern, einfache arithmetische Operationen

  Bsp.: 10110000 01100011

  (hex) B0 63

  Operand 0x63 wird in AL-Register geladen

Geschichte der Programmiersprachen

- *Sprachen der 2.Generation (2.GL):*

  Assemblersprachen:

  Operationen des Prozessors in lesbarerem Text, registerorientiert,

  symbolische Abkürzungen für (Programm-) Adressen → Sprünge und Schleifen,

  Übersetzer: Assemblerprogramm → Maschinensprache

  Bsp. in Assembler:

  ```
  mv  63h, al
  ```

Geschichte der Programmiersprachen

Assemblerprogramm:

```
                MOV     SIZE,R2
                MOV     #ARRAY,R1
                MOV     (R1)+,R0
                JSR     PC,COMP
                HALT
SIZE:           .WORD   6
ARRAY:          .WORD   137,10,205,647,53,122
COMP:           DEC     R2
                BEQ     RETURN
                CMP     (R1)+,R0
                BLT     REPEAT
                MOV     -(R1),R0
REPEAT:         JSR     PC,COMP
RETURN:         RTS     PC
                .END
```

Geschichte der Programmiersprachen

Assemblerprogramm:

```
                MOV     SIZE,R2                 //Größe des Feldes unter SIZE -> R2
                MOV     #ARRAY,R1               //Anfangsadr. des Feldes -> R1
                MOV     (R1)+,R0                //1.Feldlement nach R0 laden
                JSR     PC,COMP                 //Sprung an Marke COMP
                HALT
SIZE:           .WORD   6                       //Feldgröße wird hier festgelegt
ARRAY:          .WORD   137,10,205,647,53,122   //das Feld
COMP:           DEC     R2                      //R2 um1 erniedrigen (Restfeldgröße)
                BEQ     RETURN                  //wenn 0: Feld abgearbeitet, Rückspr.
                CMP     (R1)+,R0                //Vergleich akt. Elem mit nächstem
                BLT     REPEAT                  //((R1)-R0 < 0: Sprung zu REPEAT
                MOV     -(R1),R0                //(alten) Wert von R1 nach R0
REPEAT:         JSR     PC,COMP                 //Sprung nach COMP
RETURN:         RTS     PC                      //Zurücksetzen des Programmzählers
                .END
```

Symbolische Programmadressen

Geschichte der Programmiersprachen

- *Sprachen der 3. Generation (3. GL)*

  „höhere Programmiersprachen“ genannt,

  ab Anfang der 60ger Jahre,

  Abstraktion von konkreter Hardware: nicht länger Befehlssatz eines speziellen Prozessors

  mehrere konzeptuelle Ausrichtungen:

  – prozedurale Sprachen
  – objektorientierte Sprachen
  – funktionale Sprachen
  – logische Sprachen

  } die 4 großen Sprachparadigmen

Geschichte der Programmiersprachen

Prozedurale Sprachen:

- problemorientiert, strukturiert
- theoretische Grundlage: Von-Neumann-Maschine
- erster, am weitesten verbreiteter Vertreter:
  FORTRAN (FORmular TRANslation), ab 1954,
  zur Berechnung mathematischer/physikalischer Formeln
- ALGOL (ALGOrithmic Language), ab 1960,
  erstmals Blockstruktur
- COBOL (COmmon Business Oriented Language), ab 1960
- PASCAL, ab 1970, entwickelt von N. Wirth, strukturierte Programmierung statt „GOTOs“
- C, Modula-2, Basic, Ada,…

Geschichte der Programmiersprachen

Objektorientierte Sprachen:

- ab 1970
- erster Vertreter (Ursprung): SMALLTALK, entwickelt am Palo Alto Research Center von Xerox
- weitere: C++, Eiffel, Java, …
- seit 1990 Erweiterung vieler prozeduraler Sprachen um o.-o. Konzepte (sog. Hybride, wie z.B. C++)

prozedurale und objektorientierte Sprachen:

„imperative Sprachen“

Geschichte der Programmiersprachen

Funktionale Sprachen:

- auch als deklarative Sprachen bezeichnet
- beruhen auf dem λ-Kalkül
- Programm: nur Funktionendeklarationen, keine Handlungsabläufe
- Anwendungsbereiche: KI, Compilerbau, Computeralgebra
- erster Vertreter: LISP
- weitere: HOPE, ML, Haskell, Scheme, Miranda

Geschichte der Programmiersprachen

Logische Sprachen:

- Datenbasis bestehend aus Fakten und Regeln
- Fakten: wahre Aussagen, in Form von Prädikaten
- Regeln: Horn-Klauseln
- Resolutionsverfahren leitet neue Aussagen aus Fakten und Regeln ab
- bekanntester Vertreter: Prolog (PROgramming in LOGic), Anfang der 70ger Jahre

Logische und funktionale Sprachen werden oftmals als deklarative Sprachen bezeichnet: Problembeschreibung statt Algorithmusimplementation

Geschichte der Programmiersprachen

- *Sprachen der 4. Generation (4.GL):*
  keine speziellen Sprachparadigmen,
  oftmals Sprachen mit höherem Abstraktionsniveau,
  „4.GL“ als Marketinginstrument genutzt
  Beispiele:
  - Skriptsprachen:
    - SQL (Datenbankabfragesprache)
    - TeX, LateX (Seitenlayoutsprache)
    - awk, Perl, Tcl,….
  - deskriptive Sprachen (ohne Programmlogik):
    - XML, IDL, HTML,…..

Geschichte der Programmiersprachen

- es gibt viele hundert Programmiersprachen
- nur wenige Dutzend sind heute wichtig/von praktischer Relevanz

Sprache	**Kurzbeschreibung**
Ada	Für sicherheitskritische Anwendungen
ALGOL	Vorgänger vieler imperativer Sprachen
ANSYS	„Analysis Systems“, für Finite-Elemente-Bereich
APL	für Großrechner
APT	Sprache für Werkzeugmaschinen
Assembler	Prozessornahe Programmierung
AutoCAD	4GL-Sprache für CAD
awk	Stringverarbeitung unter UNIX

BASIC	Allzweckprogrammiersprache für Anfänger
BCL	Vorläufer von C
C	Hardwarenahe Universalprogrammiersprache
C*	C-Dialekt zur Parallelprogrammierung
C#	C-Derivat von Microsoft
C++	o.-o. Erweiterung von C
CLOS	Erweiterung von LISP um o.-o. Konzepte
CLP	Constrainted Logic Programming
COBOL	Im Bankenbereich von weit verbreitet
CSP	Communicating Sequential Processes, von Hoare
Delphi	o.-o. Programmiersprache
DSL	Vorläufer von PostScript
Dylan	Dynamic Language, Abkömmling von Scheme
Eiffel	o.-o. Programmiersprache
Erlang	Echtzeitfähige Sprache mit funktionalem Paradigma
FORTRAN	(ursprünglich) im wissenschaftlichen Bereich
Haskell	funktionale Sprache, auf λ-Kalkül und Miranda basierend
HTML	deskriptive Sprache für Web-Seiten

IDL	Interface Definition Language, abst. Beschreibg verteilter Objekte
Java	Universelle o.-o.Sprache
JavaScript	Skriptsprache, hauptsächlich zur Einbettung in HTML-Seiten
LISP	symbolorientierte Sprache, basiert auf λ-Kalkül, Einsatz: KI
Logo	LISP-Abkömmling, kindgerechte Umsetzung einer Interpreterspr.
Mathematica	programmierbares Computeralgebrasystem
MATLAB	MATrix LABoratory, math.Software für comp.gestützte Simulation
Miranda	leichte Skriptsprache, ab 1985,wird nicht mehr gepflegt
Modula-2	Erweiterung von Pascal um modulare Konzepte, 1978 von Wirth
Oberon	Nachfolger von Modula-2 und Pascal, Einsatzbereich: Lehre
Objective-C	SMALLTALK-ähnliche Erweiterung von C
Occam	Sprache zur Parallelprogrammierung von Transputern
Pascal	Prozedurale Sprache von Wirth, ab 1968, früher Hauptlehrsprache
PEARL	Sprache für Echtzeit- und Multitaskingsysteme
Perl	Universelle Skriptsprache, Grundlage vieler Webanwendungen
PHP	Hypertext Preprocessor, dynamische Erstellung von Webseiten
PL/1	Programming Language 1, Vorläufer von C
PostScript	Seitenbeschreibungssprache, von Adobe

PROLOG	Programming in Logic, von Alain Colmerauer
Python	Skriptsprache, für verteiltes BS Amoeba entworfen
Ruby	Skriptsprache, ähnlich Python, mit funkt. und o.-o. Elementen
Scheme	LISP-Dialekt
SGML	Standard Generalized Markup Lang., deskr. Spr. zur Textmarkierung
Simula	Vorläufer von SMALLTALK, 1966
SMALLTALK	objekt-orientierte Programmiersprache, von Kay, Ingalls und Goldberg
SNOBOL	Sprache zur Manipulation von Zeichenketten
SQL	Structured Query Language, Datenbankabfragesprache, seit 1980
Tcl	Tool Command Language, Skriptsprache mit Einflüssen von C u. LISP
Visual Basic	auf Basic basierende Sprache von Microsoft
VRML	Virtual Reality Modeling Language, zur Modellierung 3-dim.Welten
XML	eXtensible Markup Language, Metasprache zur Definition deskr. Spr.
XSLT	Funktionale PS zur Transformation von XML-Dokumenten

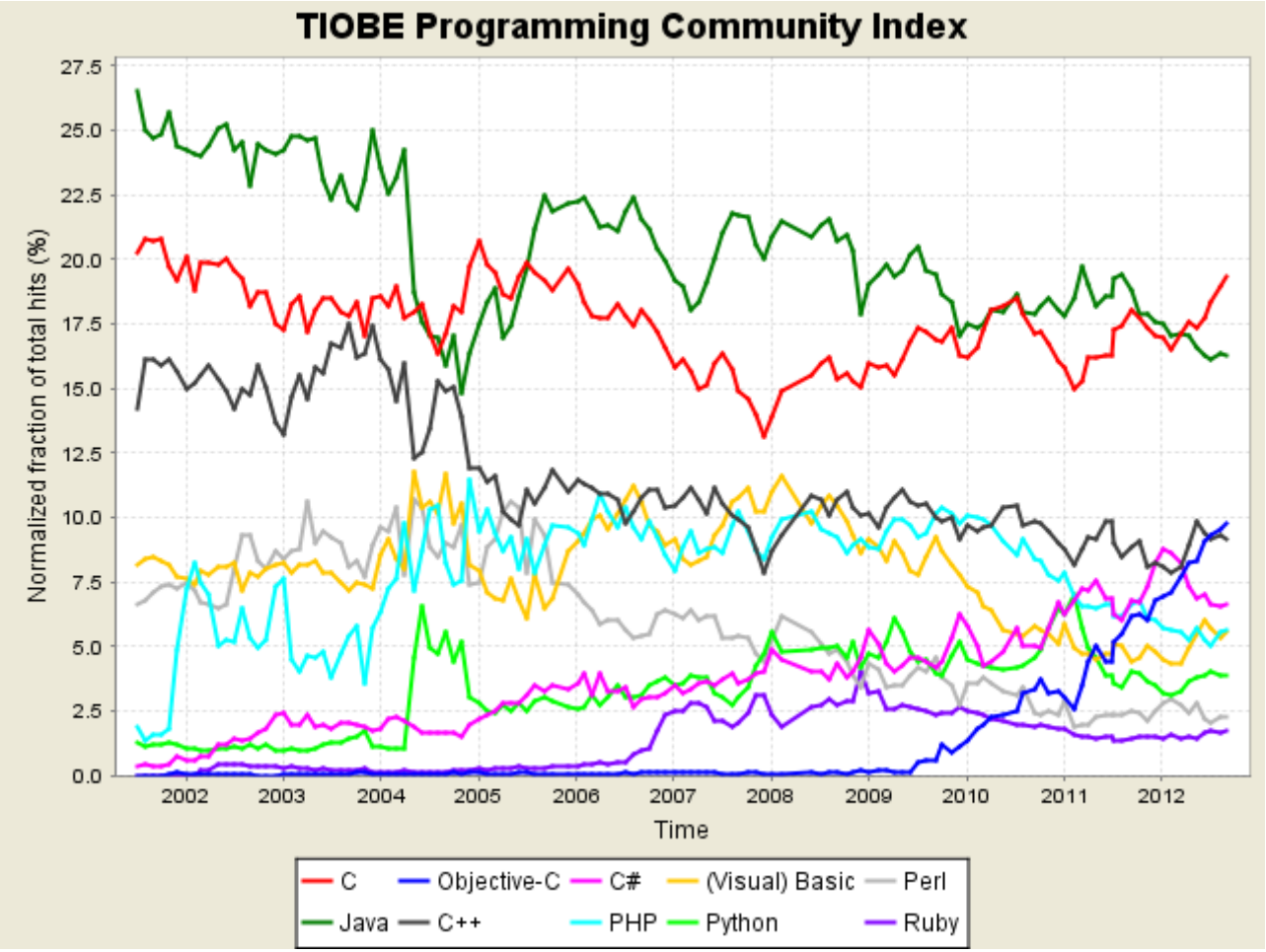
TIOBE Programming Community Index Sept. 2012

Position Sept. 2012	Position Sept. 2011	Position Sept 2010	Delta in Position	Sprache	Rating Sept 2012	Delta zu Sept 2011
1	2	2	+	C	19,295%	+1,29%
2	1	1	-	Java	16,267%	-2,49%
3	6	8	+++	Objective-C	9,770%	+3,61%
4	3	3	-	C++	9,147%	+0,30%
5	4	6	-	C#	6,596%	-0,22%
6	5	4	-	PHP	5,614%	-0,98%
7	7	5	=	(Visual)Basic	5,528%	+1,11%
8	8	7	=	Python	3,861%	-0,14%
9	9	9	=	Perl	2,267%	-0,20%
10	11	10	+	Ruby	1,724%	+0,29%
11	10	11	-	JavaScript	1,328%	-0,14%
12	12	12	=	Delphi/ObjectPascal	0,993%	-0,32%
13	14	13	+	Lisp	0,969%	-0,07%
15	39	18	--	Visual Basic .NET	0,840%	+0,53%
16	16	14	=	Pascal	0,830%	-0,02%
25		33		Cobol	0.431%	
32				Prolog	0,336%	

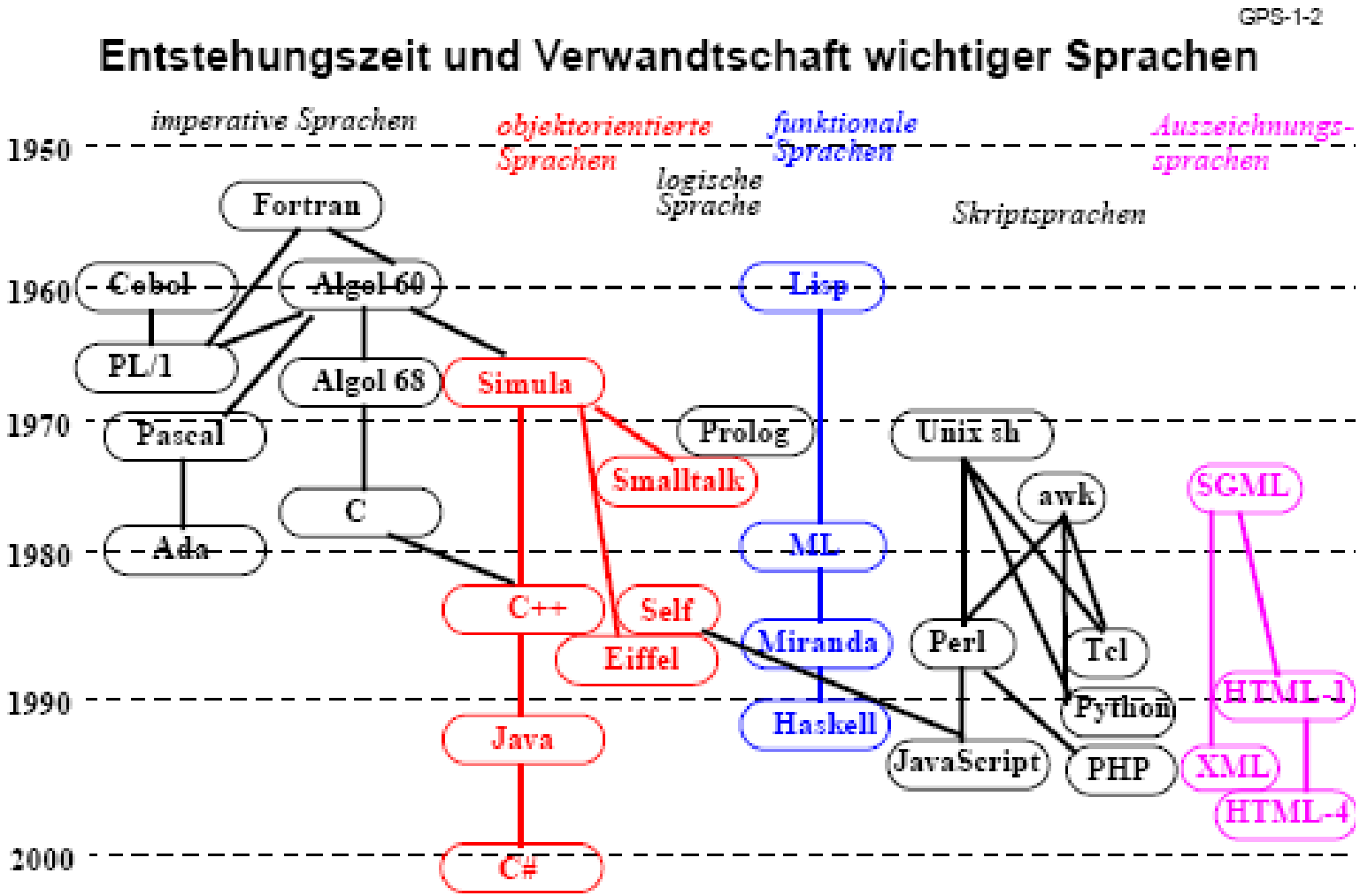
TIOBE Programming Community Langzeit-Index

Sprache	Position Sept. 2012	Position Sept. 2009	Position Sept. 1997	Position Sept. 1987
Java	2	1	5	-
C	1	2	1	1
C++	3	3	2	6
PHP	6	5	-	-
(Visual) Basic	7	6	3	5
C#	5	7	-	-
Python	8	8	29	-
Objective-C	3	44	-	-
Perl	9	4	7	-
Ruby	10	25	-	-
Lisp	13	14	10	3
Ada	18	17	16	2

TIOBE Programming Community Index
2001 - 2012



Geschichte der Programmiersprachen



Geschichte der Programmiersprachen

Ziel der Vorlesung ist es <u>nicht</u>

- herauszufinden, welches die beste oder schönste Programmiersprache ist
- die einzig wahre Programmiersprache zur Beseitigung aller anderen Programmiersprachen zu finden oder zu konstruieren
- alle Programmiersprachen zu lernen, die es gibt

sondern

- die verschiedenen Paradigmen (besser) kennen zu lernen und zu verstehen
- exemplarisch Vertreter dieser Paradigmen zu lernen

Geschichte der Programmiersprachen

Aber

Keine Programmiersprache dieser Welt macht das Programmieren letztlich einfach.

Keine Sprache kann verhindern, dass jemand darin schlechte Programme schreibt.