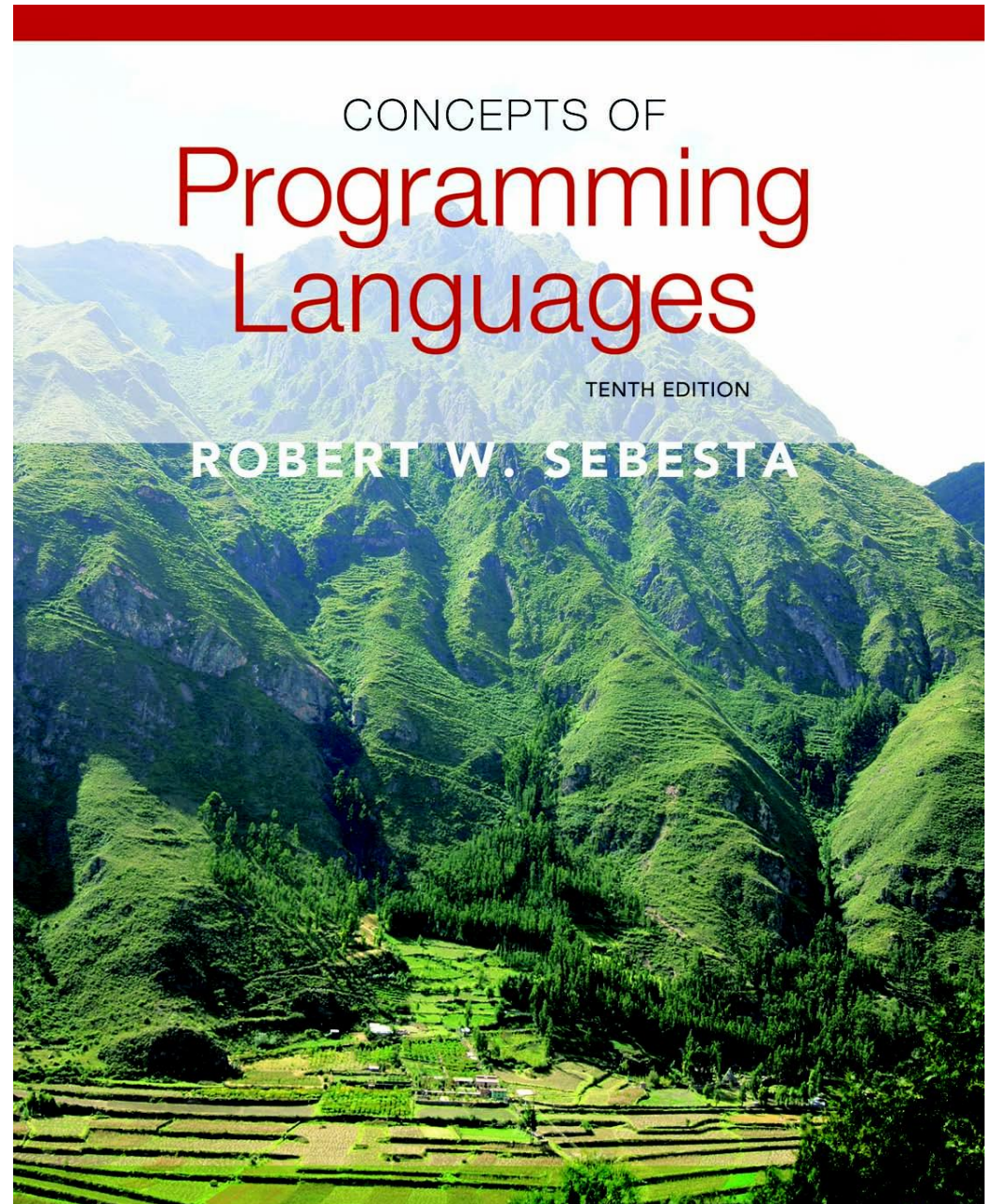


Chapter 2

Evolution of the Major Programming Languages – Part 2



The Beginning of Timesharing: BASIC

- Designed by Kemeny & Kurtz at Dartmouth
- Design Goals, (implemented in 1971):
 - Easy to learn and use for non-science students
 - Must be “pleasant and friendly”
 - Fast turnaround for homework
 - Free and private access
 - User time is more important than computer time
- Current popular dialect: Visual BASIC
- First widely used language with time sharing

2.8 Everything for Everybody: PL/I

- Designed by IBM and SHARE
- Computing situation in 1964 (IBM's point of view)
 - Scientific computing
 - IBM 1620 and 7090 computers
 - FORTRAN
 - SHARE user group
 - Business computing
 - IBM 1401, 7080 computers
 - COBOL
 - GUIDE user group

PL/I: Background

- By 1963
 - Scientific users began to need more elaborate I/O, like COBOL had; business users began to need floating point and arrays for MIS
 - It looked like many shops would begin to need two kinds of computers, languages, and support staff--too costly
- The obvious solution
 - Build a new computer to do both kinds of applications
 - Design a new language to do both kinds of applications

PL/I: Design Process

- Designed in five months by the 3 X 3 Committee
 - Three members from IBM, three members from SHARE
- Initial concept
 - An extension of Fortran IV
- Initially called NPL (New Programming Language)
- Name changed to PL/I in 1965

PL/I: Evaluation

- PL/I contributions
 - First unit-level concurrency
 - First exception handling
 - Switch-selectable recursion
 - First pointer data type
 - First array cross sections
 - i.e. 3rd row of a matrix
- Concerns
 - Many new features were poorly designed
 - Too large and too complex

Two Early Dynamic Languages: APL and SNOBOL

- Characterized by dynamic typing and dynamic storage allocation
- Variables are untyped
 - A variable acquires a type when it is assigned a value
- Storage is allocated to a variable when it is assigned a value

APL: A Programming Language

- Designed as a hardware description language at IBM by Ken Iverson around 1960
 - Highly expressive (many operators, for both scalars and arrays of various dimensions)
 - Programs are very difficult to read
 - Got notorious as “throw-away” programming
- Still in use; minimal changes

SNOBOL

- Designed as a string manipulation language at Bell Labs by Farber, Griswold, and Polensky in 1964
- Powerful operators for string pattern matching
- Slower than alternative languages (and thus no longer used for writing editors)
- Still used for certain text processing tasks

The Beginning of Data Abstraction: SIMULA 67

- Designed primarily for system simulation in Norway
- Based on ALGOL 60 and SIMULA I
- Primary Contributions
 - Coroutines – a kind of subprogram
 - Not concurrency, only one runs but can save its state
 - Classes, objects, and inheritance

Orthogonal Design: ALGOL 68

- From the continued development of ALGOL 60 but not a superset of that language
- Source of several new ideas (even though the language itself never achieved widespread use)
- Design is based on the concept of orthogonality
 - A few basic concepts, plus a few combining mechanisms

ALGOL 68 Evaluation

- Contributions
 - User-defined data structures
 - Reference types
 - Dynamic arrays (called flex arrays)
 - Does not specify subscript bounds
- Comments
 - Less usage than ALGOL 60
 - Had strong influence on subsequent languages, especially Pascal, C, and Ada

Pascal – 1971

- Developed by Wirth (a former member of the ALGOL 68 committee)
- Designed for teaching structured programming
- Small, simple, nothing really new
- Largest impact was on teaching programming
 - From mid-1970s until the late 1990s, it was the most widely used language for teaching programming

C – 1972

- Designed for systems programming (at Bell Labs by Dennis Richie)
- Evolved primarily from BCLP and B, but also ALGOL 68
- Powerful set of operators, but poor type checking
- Initially spread through UNIX
- Though designed as a systems language, it has been used in many application areas

Programming Based on Logic: Prolog

- Developed, by Comerauer and Roussel (University of Aix–Marseille), with help from Kowalski (University of Edinburgh)
- Based on formal logic
- Non–procedural
- Can be summarized as being an intelligent database system that uses an inferencing process to infer the truth of given queries
- Comparatively inefficient
- Few application areas

History's Largest Design Effort: Ada

- Huge design effort, involving hundreds of people, much money, and about eight years
- Sequence of requirements (1975–1978)
 - (Strawman, Woodman, Tinman, Ironman, Steelman)
- Named Ada after Augusta Ada Byron, the first programmer

Ada Evaluation

- Contributions
 - Packages – support for data abstraction
 - Exception handling – elaborate
 - Generic program units
 - i.e. a sort program that uses unspecified data type
 - Concurrency – through the tasking model
- Comments
 - Competitive design
 - Included all that was then known about software engineering and language design
 - First compilers were very difficult; the first really usable compiler came nearly five years after the language design was completed

Ada 95

- Ada 95 (began in 1988)
 - Support for OOP through type derivation
 - Better control mechanisms for shared data
 - New concurrency features
 - More flexible libraries
- Ada 2005
 - Interfaces and synchronizing interfaces
- Popularity suffered because the DoD no longer requires its use but also because of popularity of C++

Object–Oriented Programming: Smalltalk

- Developed at Xerox PARC, initially by Alan Kay, later by Adele Goldberg
- First full implementation of an object–oriented language (data abstraction, inheritance, and dynamic binding)
- Pioneered the graphical user interface design
- Promoted OOP

Combining Imperative and Object-Oriented Programming: C++

- Developed at Bell Labs by Stroustrup in 1980
- Evolved from C and SIMULA 67
- Facilities for object-oriented programming, taken partially from SIMULA 67
- A large and complex language, in part because it supports both procedural and OO programming
- Rapidly grew in popularity, along with OOP
- ANSI standard approved in November 1997
- Microsoft's version: MC++
 - Properties, delegates, interfaces, no multiple inheritance

Related OOP Languages

- Objective-C (designed by Brad Cox – early 1980s)
 - C plus support for OOP based on Smalltalk
 - Uses Smalltalk's method calling syntax
 - Used by Apple for systems programs
- Delphi (Borland)
 - Pascal plus features to support OOP
 - More elegant and safer than C++
- Go (designed at Google – 2009)
 - Loosely based on C, but also quite different
 - Does not support traditional OOP

An Imperative-Based Object-Oriented Language: Java

- Developed at Sun in the early 1990s
 - C and C++ were not satisfactory for embedded electronic devices
- Based on C++
 - Significantly simplified (does not include **struct**, **union**, **enum**, pointer arithmetic, and half of the assignment coercions of C++)
 - Supports *only* OOP
 - Has references, but not pointers
 - Includes support for applets and a form of concurrency

Java Evaluation

- Eliminated many unsafe features of C++
- Supports concurrency
- Libraries for applets, GUIs, database access
- Portable: Java Virtual Machine concept, JIT compilers
- Widely used for Web programming
- Use increased faster than any previous language
- Most recent version, 7, released in 2011

Scripting Languages for the Web

- Perl
 - Designed by Larry Wall—first released in 1987
 - Variables are statically typed but implicitly declared
 - Three distinctive namespaces, denoted by the first character of a variable's name
 - Powerful, but somewhat dangerous
 - Gained widespread use for CGI programming on the Web
 - Also used for a replacement for UNIX system administration language
- JavaScript
 - Began at Netscape, but later became a joint venture of Netscape and Sun Microsystems
 - A client-side HTML-embedded scripting language, often used to create dynamic HTML documents
 - Purely interpreted
 - Related to Java only through similar syntax
- PHP
 - PHP: Hypertext Preprocessor, designed by Rasmus Lerdorf
 - A server-side HTML-embedded scripting language, often used for form processing and database access through the Web
 - Purely interpreted

Scripting Languages for the Web

- Python
 - An OO interpreted scripting language
 - Type checked but dynamically typed
 - Used for CGI programming and form processing
 - Dynamically typed, but type checked
 - Supports lists, tuples, and hashes
- Ruby
 - Designed in Japan by Yukihiro Matsumoto (a.k.a, “Matz”)
 - Began as a replacement for Perl and Python
 - A pure object-oriented scripting language
 - All data are objects
 - Most operators are implemented as methods, which can be redefined by user code
 - Purely interpreted

Scripting Languages for the Web

- Lua
 - An OO interpreted scripting language
 - Type checked but dynamically typed
 - Used for CGI programming and form processing
 - Dynamically typed, but type checked
 - Supports lists, tuples, and hashes, all with its single data structure, the table
 - Easily extendable

The Flagship .NET Language: C#

- Part of the .NET development platform (2000)
- Based on C++ , Java, and Delphi
- Includes pointers, delegates, properties, enumeration types, a limited kind of dynamic typing, and anonymous types
- Is evolving rapidly

Markup/Programming Hybrid Languages

- XSLT
 - eXtensible Markup Language (XML): a metamarkup language
 - eXtensible Stylesheet Language Transformation (XSTL) transforms XML documents for display
 - Programming constructs (e.g., looping)
- JSP
 - Java Server Pages: a collection of technologies to support dynamic Web documents
 - JSTL, a JSP library, includes programming constructs in the form of HTML elements

Summary

- Development, development environment, and evaluation of a number of important programming languages
- Perspective into current issues in language design