Chapter 9: The Tower of Babel

Invitation to Computer Science, C++ Version, Third Edition

Why Babel?

- Story of Tower of Babel
  - A biblical story about people suddenly starting to speak different languages and no longer being able to communicate with each other
- Multiple programming languages
  - Each language designed for specific needs
  - One language may be better suited than others for writing certain kinds of programs

Objectives

In this chapter, you will learn about:

- Procedural languages
- Special-purpose languages
- Alternative programming paradigms

Procedural Languages

- Also called imperative languages
- A program consists of sequences of statements that manipulate data items
- The programmer devises the step by step sequence of “imperative commands”

FORTRAN

- FORTRAN is from FORMula TRANslation
- Developed in the mid-1950s by a group at IBM headed by John Backus
- First high-level programming language
- Remains an effective language for engineering applications

FORTRAN (continued)

- Designed for numerical computations
  - Allows concise mathematical notation and a number of mathematical functions
- Another goal: optimize the object code
  - External libraries of code modules that are separately compiled and used by a program

Story of Tower of Babel

A biblical story about people suddenly starting to speak different languages and no longer being able to communicate with each other

Multiple programming languages

- Each language designed for specific needs
- One language may be better suited than others for writing certain kinds of programs

FORTRAN

- FORTRAN is from FORMula TRANslation
- Developed in the mid-1950s by a group at IBM headed by John Backus
- First high-level programming language
- Remains an effective language for engineering applications

FORTRAN (continued)

- Designed for numerical computations
  - Allows concise mathematical notation and a number of mathematical functions
- Another goal: optimize the object code
  - External libraries of code modules that are separately compiled and used by a program
COBOL

- COBOL derives from COrmmon Business-Oriented Language
- Developed in 1959–1960 by a group headed by Grace Hopper of the U.S. Navy
- Designed to serve business needs such as managing inventories and payrolls
  - Better for file input than keyboard input

COBOL (continued)

- Much of a COBOL program may be concerned with formatting
  - Described by “PICTURE clauses” in the program
- COBOL programs
  - More verbose than other languages
  - Highly portable

COBOL (continued)

- COBOL programs
  - Easy to read
  - Well-suited for manipulating large data files
- Still the most widely used language

C/C++

- C
  - Developed in the early 1970s by Dennis Ritchie at AT&T Bell Laboratories
  - Originally designed for systems programming (UNIX)
  - Most widely used language for system software
  - Also used for general-purpose computing

C/C++ (continued)

- Why is C so popular
  - Relationship between C and UNIX
  - C’s efficiency
  - C is close to assembly language
  - Has high-level statements
  - Portability

Figure 9.1
User Hardware Interface and Programming Languages
C/C++ (continued)

- C++
  - Developed in the early 1980s by Bjarne Stroustrup at AT&T Bell Laboratories
  - A "superset" of C
  - One of the most popular modern "industrial-strength" languages, because of
    - Standardization
    - Object-orientation
    - A strong collection of library code

Ada

- Ada
  - Mid-1970s: Branches of the U. S. armed services started to develop a common high-level programming language
  - 1979: winner of design competition
  - Ada 95 Reference Manual
    - Current international standard exists

Ada (continued)

- Ada
  - Provides multiprocessing capability
  - Strongly object-oriented
  - Still used today in
    - Transportation industry
    - Safety monitoring systems at nuclear reactors
    - Financial and communication systems

C# and .NET

- C#
  - Introduced in June 2000
  - Many improvements in safe usage over C++
  - Shares many features with Java

C# and .NET (continued)

- Microsoft .NET Framework
  - Supports C# and other languages
  - Facilitates ease of development
    - Traditional text-based applications
    - GUI applications
    - Web-based programs

C# and .NET (continued)

- .NET programs are highly portable
- .NET programs are compiled into Microsoft Intermediate Language (MSIL)
  - MSIL is not tied to any particular platform
  - Just In Time compiler or JIT
    - Compiles MSIL code into object code on the user’s machine
Special-purpose Languages

- Designed for one specialized task
- Examples:
  - SQL
  - HTML
  - JavaScript

SQL

- SQL: Structured Query Language
- A database stores data
- Databases can be queried: the user can pose questions to the database
- SQL is the language used to frame database queries

HTML

- HTML: HyperText Markup Language
- HTML is the language used to create HTML documents
- Web page
  - An HTML document viewed with Web browser software

HTML (continued)

- An HTML document
  - Consists of text displayed on the Web page and tags
  - Tags are special characters
    - Formatting
    - Special effects
    - References to other HTML documents

```
<html>
  <head>
    <title>First Page</title>
  </head>
  <body>
    <h1>This is an H1 heading</h1>
    <ul>
      <li>This text is <b>BOLD</b> <em>and</em> this text is <i>italicized</i></li>
    </ul>
    <li>Hello</li>
    <li>First item<li>
      <li>Second item<li>
    </li/>
  </body>
</html>
```

Figure 9.4: HTML Code for a Web Page

```
<html>
  <head>
    <title>Page Generated by Figure 9.4</title>
  </head>
  <body>
    <table>
      <tr>
        <td>
          <input type='text' name='Name'></td>
        </tr>
    </table>
  </body>
</html>
```

Figure 9.5: Body of the Web Page Generated by Figure 9.4
Figure 9.6
Some HTML Tags

<table>
<thead>
<tr>
<th>HTML Tag</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>h1</td>
<td>Create H1 heading (bold with largest font size)</td>
</tr>
<tr>
<td>p</td>
<td>New paragraph</td>
</tr>
<tr>
<td>b</td>
<td>Bold</td>
</tr>
<tr>
<td>i</td>
<td>Italic</td>
</tr>
<tr>
<td>ul</td>
<td>Unordered list (bulleted list)</td>
</tr>
<tr>
<td>li</td>
<td>List item</td>
</tr>
<tr>
<td>a</td>
<td>Provides hyperlink address</td>
</tr>
<tr>
<td>table</td>
<td>Table</td>
</tr>
<tr>
<td>tr</td>
<td>Table row</td>
</tr>
<tr>
<td>td</td>
<td>Table data (item in a table)</td>
</tr>
</tbody>
</table>

JavaScript

- JavaScript is a scripting language
- Scripting language
  - A “lightweight” language that is
    - Interpreted (translated/executed, statement by statement)
  - Code fragments can be embedded in Web pages to make those pages active

Alternative Programming Paradigms

- A paradigm
  - A model or mental framework for representing or thinking about something
- The paradigm of procedural programming languages
  - A sequence of detailed instructions is provided to the computer

Alternative Programming Paradigms (continued)

- The paradigm of procedural programming languages (continued)
  - Each instruction accesses or modifies the contents of a memory location
  - Computer carries out the instructions one at a time, resulting in the solution to the problem

Alternative Programming Paradigms (continued)

- Alternative paradigms for programming languages
  - Viewing a program’s actions as
    - A combination of various transformations upon items (functional programming)
    - A series of logical deductions from known facts (logic programming)
    - Multiple copies of same subtask or multiple subtasks of same problem being performed simultaneously by different processors (parallel programming)

Functional Programming

- 1958: LISP (LISt Processing) language designed by John McCarthy at MIT
- Scheme
  - A functional programming language derived from LISP in the late 1970s
  - A functional programming language views every task in terms of functions
Functional Programming (continued)

- In a functional programming language
  - Primitive functions are part of the language
  - Other functions can be defined and named by the programmer
  - Once defined, functions can be used in the definition of other functions
- Functional programming languages sometimes called applicative languages

---

Logic Programming

- Logic programming
  - Various facts are asserted to be true
  - On the basis of these facts, a logic program can infer or deduce other facts
  - A query can be posed to the program
    - The program applies logical deductions to answer the query
- Logic programming languages are sometimes called declarative languages

---

Logic Programming (continued)

- Prolog programs
  - Consist of “facts” and “rules”
    - A fact expresses a property about a single object or a relationship among several objects
    - A rule is a declaration of an “if A then B” form
  - We interact with the program by posing queries

---

Figure 9.9
Scheme Program to Add Nonnegative Integers

```scheme
(define (adder input-list)
  (cond ((null? input-list) 0)
        (else (+ (car input-list) (adder (cdr input-list))))))
```
Logic Programming (continued)

- Logic programming paradigm
  - The program is a knowledge base of facts and rules about a certain domain of interest
  - Interaction with the program: posing queries to an inference engine (also called a query interpreter)

Parallel Programming

- SIMD (single instruction stream/multiple data stream)
  - A single control unit broadcasts a single program instruction to multiple ALUs
  - Each ALU carries out that instruction on its own local data stored in its local memory

Parallel Programming (continued)

- MIMD (multiple instruction stream/multiple data stream)
  - Numerous interconnected processors execute their own programs on their own data, communicating results as necessary
- Variations of parallel processing
  - Divide-and-conquer approach to MIMD parallel processing
  - Neural networks

Figure 9.10
A Prolog Program

president(lincoln, gettysburg_address).
president(lincoln, civil_war).
president(nixon, first_moon_landing).
president(jefferson, lewis_and_clark).
president(kennedy, cuban_missile_crisis).
president(fdr, world_war_ii).
before(jefferson, lincoln).
before(lincoln, fdr).
before(fdr, kennedy).
before(kennedy, nixon).
precedes(X, Y) :- before(X, Y).
precedes(X, Y) :- before(X, Z), precedes(Z, Y).

Figure 9.11
The Logic Programming Paradigm

Figure 9.13
An Example of MIMD Parallel Processing
Summary

- Each programming language was designed to meet specific needs
- Procedural programming languages: FORTRAN, COBOL, C, Ada, Java, C++, C#, Visual Basic
- Special-purpose languages: SQL, HTML, JavaScript
- A functional programming language views every task in terms of functions

Summary

- Logic programming: various facts are asserted to be true, based on whether the program infers or deduces other facts
- Parallel programming
  - SIMD (single instruction stream/multiple data stream)
  - MIMD (multiple instruction stream/multiple data stream)