Introduction of Programming Languages

Programming Language Concepts

- What is a programming language?
- Why are there so many programming languages?
- What are the types of programming languages?
- Does the world need new languages?

What is a Programming Languages

- A programming language is a set of rules that provides a way of telling a computer what operations to perform.
- A programming language is a set of rules for communicating an algorithm
- It provides a linguistic framework for describing computations

What is a Programming Language?

A programming language is a notational system for describing computation in a machine-readable and human-readable form.

A programming language is a tool for developing executable models for a class of problem domains.

What is a Programming Language

- English is a natural language. It has words, symbols and grammatical rules.
- A programming language also has words, symbols and rules of grammar.
- The grammatical rules are called syntax.
- Each programming language has a different set of syntax rules.

Why Are There So Many Programming Languages

- Why does some people speak French?
- Programming languages have evolved over time as better ways have been developed to design them.
  - First programming languages were developed in the 1950s
  - Since then thousands of languages have been developed
- Different programming languages are designed for different types of programs.

Levels of Programming Languages

High-level program

```
class Triangle {
  ...
  float surface()
    return b*h/2;
  }
```

Low-level program

```
LOAD r1,b
LOAD r2,h
MUL r1,r2
DIV r1,#2
RET
```

Executable Machine code

```
0001001001000101
0010010011101100
10101101001...
```

What Are the Types of Programming Languages

- First Generation Languages
- Second Generation Languages
- Third Generation Languages
- Fourth Generation Languages
- Fifth Generation Languages

First Generation Languages

- Machine language
  - Operation code – such as addition or subtraction.
  - Operands – that identify the data to be processed.
  - Machine language is machine dependent as it is the only language the computer can understand.
  - Very efficient code but very difficult to write.

Second Generation Languages

- Assembly languages
  - Symbolic operation codes replaced binary operation codes.
  - Assembly language programs needed to be “assembled” for execution by the computer. Each assembly language instruction is translated into one machine language instruction.
  - Very efficient code and easier to write.

Third Generation Languages

- Closer to English but included simple mathematical notation.
  - Programs written in source code which must be translated into machine language programs called object code.
  - The translation of source code to object code is accomplished by a machine language system program called a compiler.

Third Generation Languages (cont'd.)

- Alternative to compilation is interpretation which is accomplished by a system program called an interpreter.
- Common third generation languages
  - FORTRAN
  - COBOL
  - C and C++
  - Visual Basic

Fourth Generation Languages

- A high level language (4GL) that requires fewer instructions to accomplish a task than a third generation language.
- Used with databases
  - Query languages
  - Report generators
  - Forms designers
  - Application generators

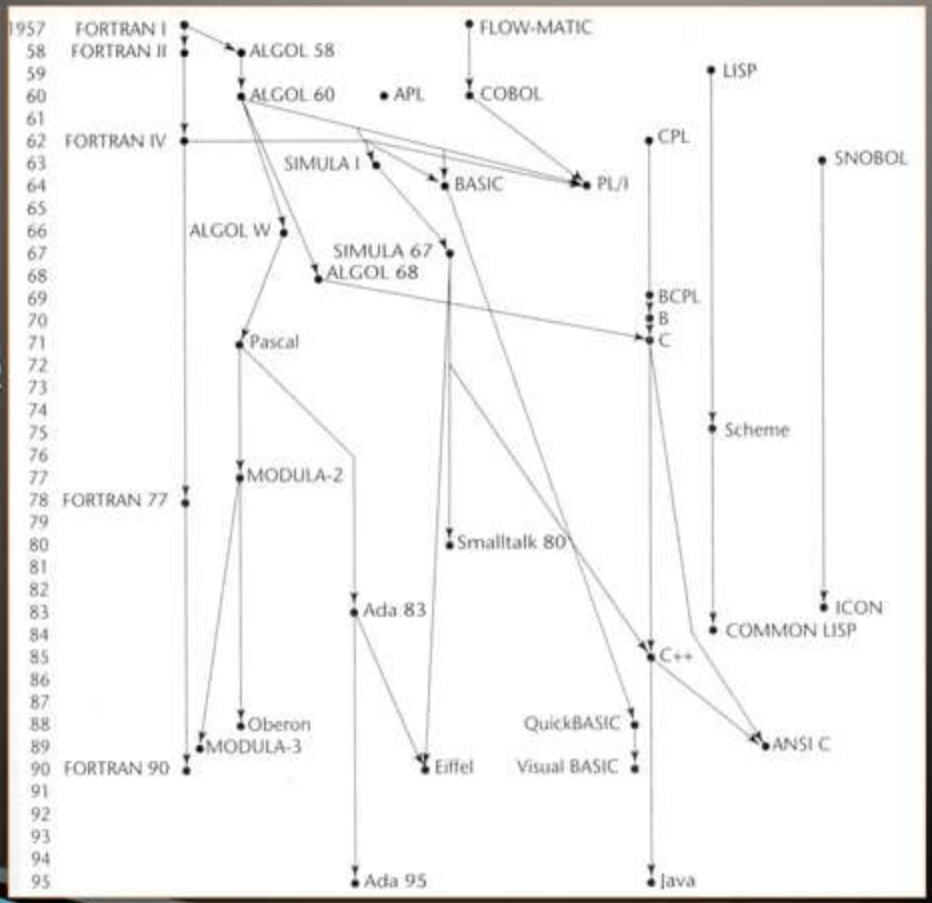
Fifth Generation Languages

- Declarative languages
- Functional(?): Lisp, Scheme, SML
  - Also called applicative
  - Everything is a function
- Logic: Prolog
  - Based on mathematical logic
  - Rule- or Constraint-based

Beyond Fifth Generation Languages

- Though no clear definition at present, natural language programs generally can be interpreted and executed by the computer with no other action by the user than stating their question.
- Limited capabilities at present.

Language Family Tree



The principal paradigms

- Imperative Programming (C)
- Object-Oriented Programming (C++)
- Logic/Declarative Programming (Prolog)
- Functional/Applicative Programming (Lisp)

Programming Languages

- Two broad groups
  - Traditional programming languages
    - Sequences of instructions
    - First, second and some third generation languages
  - Object-oriented languages
    - Objects are created rather than sequences of instructions
    - Some third generation, and fourth and fifth generation languages

Traditional Programming Languages

- FORTRAN
  - FORmula TRANslation.
  - Developed at IBM in the mid-1950s.
  - Designed for scientific and mathematical applications by scientists and engineers.

Traditional Programming Languages (cont'd.)

- COBOL
  - COmmon Business Oriented Language.
  - Developed in 1959.
  - Designed to be common to many different computers.
  - Typically used for business applications.

Traditional Programming Languages (cont'd.)

- BASIC
  - Beginner's All-purpose Symbolic Instruction Code.
  - Developed at Dartmouth College in mid 1960s.
  - Developed as a simple language for students to write programs with which they could interact through terminals.

Traditional Programming Languages (cont'd.)

- C
  - Developed by Bell Laboratories in the early 1970s.
  - Provides control and efficiency of assembly language while having third generation language features.
  - Often used for system programs.
  - UNIX is written in C.

Object-Oriented Programming Languages

- Simula
  - First object-oriented language
  - Developed by Ole Johan Dahl in the 1960s.
- Smalltalk
  - First purely object-oriented language.
  - Developed by Xerox in mid-1970s.
  - Still in use on some computers.

Object-Oriented Programming Languages (cont'd.)

- C++
  - It is C language with additional features.
  - Widely used for developing system and application software.
  - Graphical user interfaces can be developed easily with visual programming tools.

Object-Oriented Programming Languages (cont'd.)

- JAVA
  - An object-oriented language similar to C++ that eliminates lots of C++'s problematic features
  - Allows a web page developer to create programs for applications, called applets that can be used through a browser.
  - Objective of JAVA developers is that it be machine, platform and operating system independent.

Special Programming Languages

- Scripting Languages
  - JavaScript and VBScript
  - Php and ASP
  - Perl and Python
- Command Languages
  - sh, csh, bash
- Text processing Languages
  - LaTex, PostScript

Special Programming Languages (cont'd.)

- HTML
  - HyperText Markup Language.
  - Used on the Internet and the World Wide Web (WWW).
  - Web page developer puts brief codes called tags in the page to indicate how the page should be formatted.

Special Programming Languages (cont'd.)

- XML
  - Extensible Markup Language.
  - A language for defining other languages.

A language is a language is a language

- Programming languages are <u>languages</u>
- When it comes to mechanics of the task, learning to speak and use a programming language is in many ways like learning to speak a human language
- In both kind of languages you have to learn new vocabulary, syntax and semantics (new words, sentence structure and meaning)
- And both kind of language require considerable practice to make perfect.

But there is a difference!

- Computer languages lack ambiguity and vagueness
- In English sentences such as *I saw the man with a telescope* (Who had the telescope?) or *Take a pinch of salt* (How much is a pinch?)
- In a programming language a sentence either means one thing or it means nothing

What determines a "good" language

- Formerly: Run-time performance
  - (Computers were more expensive than programmers)
- Now: Life cycle (human) cost is more important
  - Ease of designing, coding
  - Debugging
  - Maintenance
  - Reusability
- FADS

Criteria in a good language design

- **Writability:** The quality of a language that enables a programmer to use it to express a computation clearly, correctly, concisely, and quickly.
- **Readability:** The quality of a language that enables a programmer to understand and comprehend the nature of a computation easily and accurately.
- **Orthogonality:** The quality of a language that features provided have as few restrictions as possible and be combinable in any meaningful way.
- **Reliability:** The quality of a language that assures a program will not behave in unexpected or disastrous ways during execution.
- **Maintainability:** The quality of a language that eases errors can be found and corrected and new features added.

Criteria (Continued)

- **Generality:** The quality of a language that avoids special cases in the availability or use of constructs and by combining closely related constructs into a single more general one.
- **Uniformity:** The quality of a language that similar features should look similar and behave similar.
- **Extensibility:** The quality of a language that provides some general mechanism for the user to add new constructs to a language.
- **Standardability:** The quality of a language that allows programs written to be transported from one computer to another without significant change in language structure.
- **Implementability:** The quality of a language that provides a translator or interpreter can be written. This can address to complexity of the language definition.