Chapter 1 – Preliminaries

* Reasons for Studying Concepts of Programming Languages
  + Increased capacity to express ideas
  + Improved background for choosing appropriate languages
  + Increased ability to learn new languages
  + Better understanding of the significance of implementation
  + Better use of languages that are already known
  + Overall advancement of computing
* Programming Domains
  + Scientific Applications- 1940s to 1950s, the first computers were invented and used for scientific applications
  + Business Applications- 1950s, special computers and languages developed for this purpose
  + AI- computer applications characterized using symbolic rather than numeric computations
  + Web Software- World Wide Web is supported by an eclectic collection of languages, such as HTML or Java
* Language Evaluation Criteria
  + Readability
    - Overall Simplicity
      * Feature multiplicity- having more than one way to accomplish an operation
      * Operator overloading- single operator symbol has more than one meaning
    - Orthogonality- relatively small set of primitive constructs can be combined in a relatively small number of ways to build the control and data structures of the language
    - Data types
    - Syntax Design
  + Writability
    - Simplicity and Orthogonality
    - Expressivity
    - Reliability
      * Type Checking
      * Exception handling
      * Aliasing- having two or more distinct names in a program that can be used to access the same memory cell
      * Readability and Writability
    - Cost
      * Portability- the ease with which programs can be moved from one implementation to another
* Influences on Language Design
  + Computer Architecture
    - Von Neumann architecture- both data and programs are stored in the same memory. CPU is separate from the memory
  + Programming Design Methodologies
* Language Categories
  + Imperative languages
  + Functional languages
  + Logic languages
  + OO languages
* Language Design Trade-offs
  + Reliability vs. Cost
  + APL- easy writability vs. poor readability
  + Writability vs. Reliability
  + Design vs. Evaluation
* Implementation Methods
  + Compilation
    - Compiler implementation- programs can be translated into machine language directly on the computer. Advantage of very fast program execution once translation process is complete
    - Source language- language that a compiler translates
    - Parse trees- hierarchical structures constructed from the syntax analyzer
    - Von Neumann bottleneck- speed of the connection between a computer’s memory and its processor
  + Pure Interpretation
  + Hybrid Implementation Systems- compromise between compilers and pure interpreters; they translate high-level language programs to an intermediate language designed for easy interpretation
  + Preprocessors- program that processes a program just before the program is compiled
* Programming Environments- the collection of tools used in the development of software