Programming Languages
Review

I. Introduction
definitions, motivation, architecture, compilation

II. PL Design Principles
  Simplicity, Semantic completeness, Abstraction, Readability, Distinct Representation, Write-
  ability, Orthogonality, Portability, Reliability, Costs, Efficiency, Design Tradeoffs

III. Evolution of PL's
  Types of PLs (Procedural, Functional, Logical, OOP, special), Generations, Programming
  Domains

IV. Natural Language and PL's
  Elements of Language, Metalanguage,

V. Formal Language Description
  Syntax, Grammars, Parse Trees, BNF, EBNF, Syntax Graphs

VI. Semantics
  A. Static (at compile time) - Attribute Grammars
  B. Dynamic Semantics
     1. Operational Semantics
     2. Axiomatic Semantics (post & preconditions)
     3. Denotational Semantics: function theory
  C. Little Quilts example
  D. Lambda Calculus
  E. Extended Semantics (e.g. FORTH)

VII. Names, Bindings, Type Checking, Scopes
  Names, variables, binding variables (types, storage), lifetime, type checking, static &
  dynamic scoping

VIII. Types
  Primitive (Numeric, Boolean, Character), User-defined, Arrays (subscripts, implementa-
  tions), Records, Unions, Sets, Pointers

IX. Expressions & Assignment
  Arithmetic, Precedence, order of evaluation, Conditionals, Overloaded operators, type con-
  versions, short-circuit evaluation, Assignment

X. Control Structures
  Compound Statements, Selection Statements, Multiple selectors, Iterative statements, Logi-
  cally controlled loops, Unconditional branches, concurrent programming

XI. Subprograms
  Parameters (modes, pass by value, value-result, reference, name), subprograms as param-
  eters

XII. Implementing Subprograms
  Fortran, ALGOL, ARI's, Recursion, Static chaining, displays

XIII. PROLOG and LISP