Principles of Programming Languages

Review:

Formal Languages, Functional Programming, and Scope

Professor Lou Steinberg

Fall 2004

Formal Languages

- Formal Language
  - Set of strings of characters: {“a”, “aa”, …}
  - Formal definition of the set: “all strings with one or more ‘a’s”
- Uses
  - Reason about computation and mechanisms for computation
  - Define programming languages

Defining Languages

- By a grammar
  - Rules: non-terminals, terminals
  - BNF, EBNF
  - Parse trees
- By an automaton
  - States, transitions, additional memory (e.g. stack, tape)
  - Start state, accepting states
  - Deterministic vs Non-deterministic
- By a Regular Expression
- Grammar <=> Machine

Classes of Languages

- Class of language <=> class of grammar <=> class of automaton
  - Regular Languages
  - Context Free Languages
- Ease of specifying / parsing vs range of languages that can be specified

Programming Languages

- Note anthropomorphism
- Language specified formally
  - syntax more so than semantics
  - want syntactic structure to match semantic structure
    - expressions
    - control structure
    - variable scope
- Expression precedence and associativity in a grammar

Programming Languages

- Goals:
  - Easy to create correct programs:
    - easy to read, write, learn
    - locality, abstraction
  - Easy to execute efficiently

Functional Languages

- A program is an expression to be evaluated
  - No side effects (e.g. assignment)
  - Variables are names for values (not places)
  - You can treat functions as data
    - create, pass as function arguments and return as values
- Advantage:
  - Simpler to reason about
  - Allows abstractions, eg map, foldr

Scheme

- Linked lists (a b), car, cdr, cons
- Control structures: if, cond, recursion, tail recursion
- Lambda, let, and define
- Closures
- Higher order functions
  - functions as arguments, results
  - functions stored in lookup tables
  - map, foldr

Scope and Memory

- Purpose: locality
- Block structure
- Free vs bound variables
- Dynamic vs lexical scope
- Stack frames, conrol and binding links, display
- Stack vs heap
- garbage collection vs explicit free