CS430 Programming Languages
Exam 2 Study Guide

Format

The exam will include short answer, multiple-choice, and/or true/false questions. Some questions may require the interpretation of code (pseudocode or some language that we have studied), but no extensive coding problems will be included. Exam questions will be similar to homework and class activity questions and problems.

Content

The exam will cover the following learning objectives:

- Chapter 16: Declarative Languages
  - Explain the formal basis for logic languages and for rule-based languages.
  - List and explain the differences between declarative and imperative languages, including their design objectives, computational model, and implementation.
  - Evaluate Prolog expressions.
  - Describe typical applications of declarative languages, and explain their strengths and weaknesses.

- Chapter 5: Variables
  - List and describe the essential attributes of a variable.
  - Explain and evaluate language options regarding variable naming
  - Explain and evaluate strong vs. weak typing
  - Describe binding times and the things that are bound at various times
  - Explain and evaluate static vs. dynamic binding in general.
  - Explain and evaluate alternatives related to storage location and lifetime.
  - Differentiate between variable lifetime and scope.
  - Define scope and scoping rules, and interpret a program based upon scoping alternatives.

- Chapter 6: Data Types
  - Describe the history of primitive data types (integer, real, decimal, Boolean), and the languages in which they originated.
  - Describe and evaluate the storage methods used for strings.
  - Explain and evaluate choices related to user-defined ordinal types.
  - Explain and evaluate array implementation choices, including subscripting, storage order, jagged arrays, and slices.
  - Describe and evaluate mechanisms used for implementing records and unions.
  - Explain and evaluate pointers vs. reference types.
  - Define dangling pointer and memory leak, and explain their causes and consequences
  - Explain and apply algorithms for heap management, including tombstones, locks-and-keys, reference counters, and garbage collection.