CISC 3160 [24] Programming Languages
4 hours; 4 credits

The design, implementation, and evolution of programming languages. Language features and their effects upon translation and run-time environments. Languages studied are chosen for their historical and current significance, programming paradigm, and run-time environment. Syntax and semantic specification; formal grammars.

Objectives:

Upon completing the course, the student will:

a. be able to describe the salient characteristics of several language paradigms (procedural, object-oriented, imperative, declarative/logic, functional).
b. understand the concept of data binding and its effect upon the semantic level of the language.
c. be familiar with standard mechanisms of realizing language semantics at execution time.
d. understand the spectrum of source-to-executable language translation, its effect upon efficiency and expressivity the corresponding relation to data binding.
e. be able to use formal techniques (such as BNF) in the specification language syntax.
f. be able to recognize the relationship between the semantic level of the language and its expressivity, efficiency, control mechanisms, and data types.
g. be able to apply the conceptual material covered in this course (i.e., binding times, run-time support etc.) to the analysis of specific languages.
h. be able identify the core semantics of data types and control constructs and to recognize the similarity and differences between data and control representations of various languages.
i. be able to code small programs that illustrate the core semantics of each of set of languages that represent the paradigms covered in the course.
j. be able to discuss the technological, software-engineering, and educational issues that propelled the evolution of programming languages.

Syllabus

I. Introduction and Brief History Chapter 1
II. Syntax and Semantics Chapter 2
III. Imperative Programming Paradigm Chaps. 3, 4, 5
IV. Object-Oriented Paradigm Chaps. 6-7.3
V. Functional Paradigm Chaps. 8, 10
VI. Logic Paradigm Chapter 11

Textbook:
Addison-Wesley Publishing, 1996

Bibliography:

- Programming Language Pragmatics, by Michael L. Scott
- Compilers, by Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman
- Structure and Interpretation of Computer Programs, by Harold Abelson and Gerald Jay Sussman