CS 3060: PROGRAMMING LANGUAGES

Semester Hours: 3.0
Contact Hours: 3
Coordinator: Robert Green
Text: Seven Languages in Seven Weeks: A Pragmatic Guide to Learning Programming Languages
Author(s): BRUCE A. TATE
Year: 2010

SPECIFIC COURSE INFORMATION

Catalog Description:
A survey of programming languages that covers multiple programming features and paradigms. The goal of this course is to expose the student to a wide variety of programming languages, paradigms, features, and syntaxes through exposure to theory and hands-on exercises. Topics covered include static, dynamic, strong, and weak typing, compiled and interpreted, object-oriented, functional, and procedural programming, and decision constructs. Prerequisite: Grade of C or better in CS 2020.

Course type: REQUIRED

SPECIFIC COURSE GOALS

- I can identify differences and similarities across programming languages
- I can describe the differences between different programming paradigms
- I can implement basic algorithms using different programming paradigms
- I can explain the strengths and weaknesses of different type systems
- I can explain the differences between program compilation and interpretation

STUDENT OUTCOMES ADDRESSED BY THIS COURSE

- B.1 Analyze a given problem, and identify and define the computing requirements appropriate to its solution
• B.2 Use current techniques, skills, and tools in computing practice

• B.3 Apply mathematical foundations, algorithmic principles, and computer science theory as appropriate in modeling and solving real-world problems

• B.5 Apply design and development principles in the construction of software systems of varying complexity

LIST OF TOPICS COVERED

• Ruby, Python, Io, Prolog, Scala, Erlang, Haskell, Clojure, Lua, Factor, Elm, Elixir, Julia, miniKanren, Idris

• Encapsulation, Data Abstraction, Polymorphism, Inheritance, Higher Order Functions, Purity (Side Effect Free), Immutability, Referential Transparency, Lazy Evaluation, Recursion, Lambda Calculus, Static Typing, Dynamic Typing, Garbage Collection, Interpreted vs. Compiled Languages