Computer Science and Engineering CSE 4713
Programming Languages

CREDIT/CONTACT HOURS: Credit Hours: 3, Contact Hours: 45

COORDINATOR:
Dr. Donna Reese/Dr. Andrew Watkins

a. Supplemental Material: No required material

SPECIFIC COURSE INFORMATION:
Catalog Description: Three hours lecture. An introduction to programming language specification and analysis. Additional topics include control structures, data types, and structures, run-time environments, binding strategies, compilers, and interpreters.
a. Prerequisites: Grade of C or better in ECE 3724 and CSE 3813
b. Required/Elective:
   Computer Science: Required
   Software Engineering: Elective
   Computer Engineering: Elective

SPECIFIC GOALS OF THE COURSE:
a. Specific Outcomes of Instruction: The major goal of this course is to develop a view of programming languages not only as means to solve problems but as subjects of analysis and design. To achieve that goal, we will take a broad look at a variety of programming language implementation issues, methods, and implications. Next, we will focus on some specific programming language design features and issues, and examine their implications. Finally, we will investigate the functional, logic, and major imperative paradigms and languages as alternatives to the object-oriented ones.

   It is expected that, by the end of the course, the student will be able to:
   1. effectively apply concepts from prerequisite courses, especially formal languages and architecture courses, in the context of evaluating the features of programming languages.
   2. explain and evaluate design and implementation features of programming languages.
   3. apply conceptual knowledge of the syntax of languages, as well as the design of language data structures and control statements, to the efficient implementation of a working language.
   4. test his or her implementation solution sufficiently well to have a reasonable level of confidence that the solution will survive unknown tests which will be applied to it
   5. demonstrate comprehension of short programs written in functional and logic paradigms
b. Criterion 3 Outcomes:
Note: Parenthesized list indicates the ABET EAC and CAC outcomes addressed by each performance criteria.

1. The student will be able to apply concepts from prerequisite courses, especially formal languages and architecture courses, effectively in the context of evaluating the features of programming languages (EAC: a; CAC: a,j)

2. The student will be able to explain and evaluate design and implementation features of programming languages. (EAC: g; CAC: f)

3. The student will be able to apply conceptual knowledge of the syntax of languages, as well as the design of language data structures and control statements, to the efficient implementation of a working language. (EAC: a, b,c; CAC: a,j,k)

4. The student will be able to test his or her implementation solution sufficiently well to have a reasonable level of confidence that the solution will survive unknown tests which will be applied to it (EAC: b,c; CAC:c,k)

5. The student will demonstrate comprehension of short programs written in functional and logic paradigms (EAC: k; CAC: i)

TOPICS COVERED:  
(Number of class hrs)

1. Introduction; history of programming language development; review of formal languages and automata, representing language syntax and semantics, application to design and implementation of modern languages. 13
2. Binding, type checking, scope, lifetime, referencing environments; design and implementation of primitive data types, structured data types, dynamically allocated space 7
3. Expressions, assignment statements, and flow-of-control statements, contrasted and compared across languages. 4
4. Issues related to procedures and functions: Implementation of dynamic and static scope, semantics of subprogram call/return paradigms, design and implementation of subprogram call/return paradigms. Runtime stack, activation records. 11
5. Functional and logic programming. 6
6. Other issues: I/O, error handling, concurrent programming, etc. 4