Knight Foundation School of Computing and Information Sciences

Course Title: Principles of Programming Languages **Date:** 3/22/2019

Course Number: COP 4555

Number of Credits: 3

Subject Area: Foundations	**Subject Area Coordinator:** Hadi Amini **email:** amini@cs.fiu.edu
Catalog Description: A comparative study of several programming languages and paradigms. Emphasis is given to design, evaluation and implementation. Programs are written in a few of the languages.	
Textbook: None	
References: Expert F#, by Don Syme, Adam Granicz, and Antonio Cisternino (Apress, 2007)	
Prerequisites Courses: COP 3530	
Corequisites Courses: None	

Type: Elective for CS (Foundations group)

Prerequisites Topics:

- Familiarity with programming in Java or C++.
- Familiarity with basic techniques of algorithm analysis.
- Familiarity with recursive algorithms.
- Familiarity with linked data structures such as linked lists and binary trees.
- Basic mathematical maturity.

Course Outcomes:

1. Master programming a functional language.
2. Master programming with recursion.
3. Be familiar with the use of context-free grammars to specify programming language syntax and with recursive descent parsing.
4. Be familiar with natural semantics for imperative and functional programming languages and their use in building interpreters.
5. Be familiar with polymorphic type systems and type inference.
6. Be familiar with issues in the design and implementation of programming languages, such as lexical versus dynamic scoping and static versus dynamic type checking.

Knight Foundation School of Computing and Information Sciences
COP 4555
Principles of Programming Languages

Relationship between Course Outcomes and Program Outcomes

BS in CS: Program Outcomes	Course Outcomes
a) Demonstrate proficiency in the foundation areas of Computer Science including mathematics, discrete structures, logic and the theory of algorithms	3, 4, 5
b) Demonstrate proficiency in various areas of Computer Science including data structures and algorithms, concepts of programming languages and computer systems.	1, 2, 3, 4, 5, 6
c) Demonstrate proficiency in problem solving and application of software engineering techniques	1, 2
d) Demonstrate mastery of at least one modern programming language and proficiency in at least one other.	1, 2
e) Demonstrate understanding of the social and ethical concerns of the practicing computer scientist.	
f) Demonstrate the ability to work cooperatively in teams.	
g) Demonstrate effective communication skills.	

Assessment Plan for the Course & how Data in the Course are used to assess Program Outcomes

Student and Instructor Course Outcome Surveys are administered at the conclusion of each offering, and are evaluated as described in the School's Assessment Plan: https://abet.cs.fiu.edu/csassessment/

Knight Foundation School of Computing and Information Sciences
COP 4555
Principles of Programming Languages

Outline

Topic	Number of Lecture Hours	Outcome
1. Functional Programming 1.1. Checklist for programming with recursion 1.2. First-class functions, currying 1.3. Efficiency: avoiding recomputation 1.4. Polymorphic types, Value restriction 1.5. Discriminated union types	18	1, 2, 5
2. Programming Language Syntax 2.1. Context-free grammars 2.2. Parse trees 2.3. Ambiguity 2.4. Recursive descent parsing	3	3
3. Programming Language Semantics 3.1. Natural Semantics 3.2. Interpreters 3.3. Expressions, commands, declarations 3.4. Variables, L-values, R-values 3.5. Environments, stack frames, lifetime, tail recursion	13	1, 4, 6
4. Types 4.1. Dynamic type checking 4.2. Static type checking 4.3. Type Inference	6	5

Course Outcomes Emphasized in Laboratory Projects / Assignments

Outcome	Number of Weeks
1	4
2	2
3	2
4	2
5	2
6	1

Oral and Written Communication

No significant coverage

Written Reports		Oral Presentations	
Number Required	Approx. Number of pages	Number Required	Approx. Time for each
0	0	0	0

Social and Ethical Implications of Computing Topics

No significant coverage

Topic	Class time	student performance measures

Approximate number of credit hours devoted to fundamental CS topics

Fundamental CS Area	Core Hours	Advanced Hours
Algorithms:		
Software Design:		
Computer Organization and Architecture:		
Data Structures:		
Concepts of Programming Languages		**3.0**

Theoretical Contents

Topic	Class time
Formal semantics	6 hours
Polymorphic type systems	6 hours

Problem Analysis Experiences

1.

Solution Design Experiences

1. Interpreter
2. Type inference

The Coverage of Knowledge Units within Computer Science Body of Knowledge[1]

Knowledge Unit	Topic	Lecture Hours
PL3. Introduction to language translation	2, 3	6
PL4. Declarations and types	1, 4	6
PL5. Abstraction mechanisms	1, 3	6
PL6. Functional programming	1	6
PL9. Type systems	1, 4	6
PL10. Programming language semantics	3	5
PF4. Recursion	1	5

[1]See https://www.acm.org/binaries/content/assets/education/cs2013_web_final.pdf for a description of Computer Science Knowledge units