Master Course Syllabus

School of Engineering and Computer Science
Washington State University Vancouver

CS 355

Programming Language Design

3 Semester Hours
(3 lecture hours)

Catalog Description

Design concepts of high-level programming languages; survey of existing languages, experience using some languages.

Prerequisite Courses

- CS 223 with a C or better
- CS 224 with a C or better
- Certified major in Computer Science

Prerequisite Topics

- Proficiency in at least one imperative programming language.
- Experience in editing, compiling/linking, executing, testing, and (source level) debugging of moderately sized programs.
- Experience with multiple source file programs
- Experience with common data structures including lists, trees, stacks, queues, graphs, hash tables, etc.

Measured Course Outcomes

Students taking this course will:

1. Evaluate and classify programming languages according to the design paradigm and features provided. (Contributes to performance criteria 1-c.)
2. Implement lexer and syntax parser for a given language. (Contributes to performance criterion 6-c.)
3. Write a program in functional language or in a logic programming language using the language's intended paradigm. (Contributes to performance criterion 6-d.)

Covered Course Outcomes

1. Design lexical and syntax analysis algorithm for a given language. (Contributes to performance criterion 6-b.)

Recommended Textbooks

Kenneth C. Louden and Kenneth A. Lambert, *Programming Languages: Principles and Practices*. Course Technology, Cengage Learning.

Reference Material

None

Major Topics Covered in the Course

1. History and evolution of programming language design criteria
2. Programming language paradigms including logic programming, functional programming, object-oriented programming, and parallel programming
3. Syntax
4. Semantics
5. Data types and abstract data types
6. Expressions, statements, procedures, and environments

Projects

The students will be given several programming assignments which require using various programing languages to complete the assignments

Programming Project Area	Weeks
Lexical and syntax analysis	2
Object oriented programming	2

Design, Implementation and Analysis

The main areas where students are involved in problem analysis are:

- Formal language description and translation.
- Abstractions for programming "in the large."
- Evaluation of programming languages in terms of their suitability for specific tasks.

A variety of techniques, and language features are presented for solving problems in the arena of programing language design, such as:

- Top down predicative parsing techniques for syntax analysis.
- Program abstraction techniques for writing and maintaining large programs.
- Reference counting and automatic garbage collection for robust memory management.

CS2013

This course provides coverage of CS2013 knowledge areas. Values listed are minimum course hours dedicated to the topic, percentages indicate the fraction of CS2013 knowledge area topics covered (acceptable values are: <25%, 25-75%, >75%, or 100%).

Area	Tier 1	Tier 2	Elective
AR/Machine Level Representation of Data		2 (25-75%)	
AR/Assembly Level Machine Organization		3 (25-75%)	
PL/Object-Oriented Programming	4 (100%)	2 (25-75%)	
PL/Functional Programming	3 (100%)	4 (100%)	
PL/Basic Type Systems	1 (100%)	3 (25-75%)	
PL/Language Translation and Execution		3 (>75%)	

Area	Tier 1	Tier 2	Elective
PL/Syntax Analysis			3 (100%)
PL/Compiler Semantic Analysis			3 (75%)
PL/Runtime Systems			1 (25-75%)
PL/Type Systems			2 (25-75%)
PL/Formal Semantics			3 (<25%)
PL/Logic Programming			2 (25-75%)

Course Coordinator:	Paul Bonamy
Last Updated:	October 14, 2020
Syllabus Version Number:	2.1