

COURSE OUTLINE OF RECORD

Dept., Number	CSC 4340	Course Title	Organization of Programming Lang.
Semester Hours	3		
Year	2006	URL (if any):	

Current Catalog Description:

This course covers formal and practical study of the definition, applications and implementation of programming languages. It includes linguistic concepts of syntax and semantics, translation of high-level languages into executable form, data structuring, sequencing features, and modularization features of representative languages. Prerequisite: CSC 2331.

Textbook:

Concepts of Programming Languages, 7th edition Robert W. Sebesta; Pearson Education (Prentice Hall).

Course Goals:

Upon completion of this course, the student should be able to:

1. Learn an unfamiliar programming language by systematically investigating its definition, application area and implementation.
2. Identify and use the most appropriate language or language feature to solve a particular problem.
3. Appreciate the relationship between language definition and its execution time characteristics.

Prerequisites by Topic:

In-depth knowledge of a higher-level language, such as C++; working knowledge of data structures and related algorithms.

Major Topics Covered in the Course (number of weeks):

Chapter/approx. number of weeks

- | | |
|---|-----|
| 1. Professional Ethics | 0.5 |
| 2. Formal Grammars | 3.5 |
| 3. Describing syntax and semantics | 2 |
| 4. Names, bindings, type checking, and scopes | 1 |
| 5. Data types | 1 |
| 6. Expressions and the assignment statement | 1 |
| 7. Statement-level control structures | 1 |
| 8. Subprograms | 1 |
| 9. Subprogram implementation | 1 |
| 10. Data abstraction | 1 |
| 11. Object-oriented programming | 1 |

Laboratory Projects :

Finite State Machine
 Lexical Scanner
 Shannon's Entropy Measure
 Precedence Grammar
 Turing Machine

Dept., Number	CSC 4340	Course Title	Organization of Programming Lang.
----------------------	----------	---------------------	-----------------------------------

Estimate Curriculum Category Content (Semester hours)

Area	Core	Advanced	Area	Core	Advanced
Algorithms			Data Structures		
Software Design	1		Prog. Languages		2
Comp. Arch.					

Oral and Written Communication:

The instruction in this course is based on class lectures, reading assignments, home or lab assignments, and exams. Typically, every student is required to submit at least 10 written assignments of 2 to 4 pages. Every student is expected to answer questions in class on these assignments, to present and explain their own solutions, and to answer questions and participate in discussions on the whole course material.

Social and Ethical Issues:

Ethical issues are covered as a case study of plagiarism. This case is interesting to students because it directly involves the textbook for this course. An author has submitted and published an article in the ACM SIGPLAN Notices that essentially paraphrases contents of the textbook. As a home assignment, students are asked to read the article and create a list of correspondence between it and the textbook. Students are also asked to discuss ethical and legal responsibilities in a case of this sort. After the assignment is graded, the whole topic is discussed in class. The class time spent on this topic is about 20 minutes but the time spent working on the assignment is significantly more.

Theoretical Content:

This course covers theoretical aspects of the definition, application, and implementation of programming languages, including formal methods for syntax description and foundations of functional and logic programming.

Problem Analysis:

The general principles of programming languages are explored through analysis of their manifestation in specific programming languages. Students are expected to analyze and evaluate various programming language features and their relationships with the other features of a language. Students are also expected to analyze concrete example programs and to interpret them in terms of the studied language features.

Open-Ended Design:

This course covers influences on language design and common design trade-offs. Most of the course is devoted to design issues of the primary features of imperative programming languages. In each case, the design choices for several example languages are presented and evaluated.