

COURSE DESCRIPTION

Dept., Number	CS440	Course Title	Programming Languages and Translators
Semester hours	3	Course Coordinator	Dr. Xiang-Yang Li, Associate Professor

Current Catalog Description

Study of commonly used computer programming languages with an emphasis on precision of definition and facility in use. Scanning, parsing, and introduction to compiler design. Use of compiler generating tools. Prerequisite: (CS 330 and CS 351) or CS401 or CS403. (3-0-3) (T)

Textbook

<http://dijkstra.cs.iit.edu/cs440-sp08/resources/>

References

Course Outcomes

Students should be able to:

- Explain major classes of programming languages: techniques, features, and styles.
 - Know how to use boxed and unboxed variables
 - Be able to use higher order functions.
- How to specify formally the meaning of a language --- to people and to the computer.
 - Use Transition, Typing, and Denotational Semantics to define a language construct.
 - Be able to specify the language of regular expressions.
 - Determine if a grammar is LL, and write a parser for it using recursive descent.
 - Determine if a grammar is LR, and write a parser for it using a parser generator.
 - Describe the algorithm for both LL and LR parser generation.
- Explain Three Powerful Ideas:
 1. Recursion
 - Know how to use both tail recursion and standard recursion.
 - Know how to use higher order functions to eliminate recursion.
 2. Abstraction
 - Know how to create user-defined types.
 - Know how to use functions to model integers.
 - Know how to use trees to model language constructs.

3. Transformation

- Know how to interpret a language.
- Know how to use unification.
- How to choose a language.
- How to implement a language.

Emphasis: learn theory and apply it.

Relationship between Course Outcomes and Program Outcomes

The following Program Outcomes are supported by the above Course Outcomes:

- a. An ability to apply knowledge of computing and mathematics appropriate to the discipline
- c. An ability to design, implement and evaluate a computer-based system, process, component, or program to meet desired needs
- h. Recognition of the need for, and an ability to engage in, continuing professional development
- i. An ability to use current techniques, skills, and tools necessary for computing practices.
- j. An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices
- l. Be prepared to enter a top-ranked graduate program in Computer Science.

Prerequisites by Topic

Experience writing basic programs in more than one computer language and a strong discrete mathematics background.

Major Topics Covered in the Course

1. Course Introduction, Recursion, User Defined Types, Higher Order Functions, Interpreters	7.5 hours
2. Regular Languages, Grammars, LL Parsing, LR Parsing, LR Parsing Tools, Lambda Calculus	9 hours
3. Unification, The Call Stack and the Heap, Transition Semantics, Natural Semantics, Type Semantics	7.5 hours
4. Variables, Parameters, Local State, Objects, Infinite Data, Continuation-Passing Style	9 hours
5. Prolog, Prolog's Cut Operator, Dynamic Prolog, Applications of Prolog	6 hours
6. Meta-Programming	3 hours
Midterm Exams	3 hours
Final Exam	-

45 hours

Assessment Plan for the Course

End of every semester Course Objective Assessments by CS department. End of semester Course Evaluations by IIT. Reviewed every Spring semester by CS Undergraduate Studies Committee for possible updates in the following Fall. Once every 4-5 years a detailed review of all materials for the course is made by the CS Undergraduate Studies Committee.

How Data in the Course is Used to Assess Program Outcomes (unless adequately covered already in the assessment discussion under Criterion 4)

See the assessment discussion under Criterion 4

For a computer science program

Estimate Curriculum Category Content (Semester hours)

Area	Core	Advanced	Area	Core	Advanced
Algorithms			Software design		
Data structures			Concepts of programming languages		3