

COURSE DESCRIPTION

Department and Course Number: COMP 6230

Course Title: Declarative Programming Languages and Principles

Total Credits: 3

Required: No

Prerequisites: COMP 3220

Class meetings per week: 3 hours

Lab meetings per week: 0 hours

Course Coordinator: Dr. Homer Carlisle

Date Prepared: February 15, 2004

Current Catalog Description:

Functional and logic programming theoretical foundations, models and implementation issues; example language studies.

Textbooks:

Kryisia Broda, Susan Eisenbach, Hessam Khoshnevisan and Steve Vickers. 1995. Reasoned Programming. Prentice Hall. ISBN 0-13-098831-6.

References:

1. Haskell – <http://www.haskell.org/>
2. Prolog – <http://www.swi-prolog.org>
3. Mozart – <http://www.mozart-oz.org>

Course Objectives:

1. Gain an in-depth understanding of declarative programming principles, languages and programming techniques.

Prerequisites by Topic:

1. Introduction to programming language paradigms and examples

Topics Covered: (specify number of hours on each)

1. Programming with functions mathematical foundations (6 hours)
2. Functional programming basics (3 hours)
3. Higher order functions (3 hours)
4. Evaluation modes (3 hours)
5. Example functional languages; problem solving (6 hours)
6. Implementation issues (6 hours)
7. Type checking and inference, reduction, combinators, dataflow, garbage collection, Programming with logic, mathematical foundations (6 hours)
8. Propositional logic, predicate logic, logic programming techniques (3 hours)
9. Cuts and tail recursion, logic programming and functional logic, logic programming paradigms (3 hours)
10. Forward and backward chaining, example logic programming languages; problem solving (4 hours)

11. Exams (2 hours)

Laboratory Projects: (specify number of weeks on each)

1. Haskell assignment 1 (1 week)
2. Haskell assignment 2 (2 weeks)
3. Haskell assignment 3 (2 weeks)
4. Prolog assignment 1 (1 week)
5. Prolog assignment 2 (2 weeks)
6. Mozart assignment 1 (1 week)
7. Mozart assignment 2 (2 weeks)

Oral and Written Communications:

None.

Social and Ethical Issues:

None.

Theoretical Content:

Mathematical foundations of declarative computing.

Problem Analysis and Solution Design:

Students apply the analysis and design skills already acquired to the development of software components in various languages. Each component has stated requirements and students are responsible for applying a controlled, iterative process for development. The requirements for some components are relatively straightforward and are designed to illustrate particular language paradigms while the requirements for other components are non-trivial and involve significant problem analysis and solution design.

