**CSC 1209 Logic Programming**

(a) Description

This course introduces a paradigm where computation arises from proof search in logic according to a fixed, predictable strategy. It thereby unifies logical specification and implementation in a way that is quite different from functional or imperative programming. This course pro- vides a thorough, modern introduction to logic programming. It introduces the basic concepts and techniques of logic programming followed by successive refinement towards more efficient implementations or ex- tensions to richer logical concepts. It covers a variety of logics and operational interpretations.

(b) Aims

The aim of the course is to provide a basic introduction to the logic programming language, Prolog. It aims at introducing a number of logical systems of importance in computer science.

(c) Learning outcomes

By the end of the subject, students should:

*•* Be conversant with the syntax and semantics of propositional and predicate logic

*•* Be familiar with a variety of applications of predicate logic in software verification, databases and knowledge-based systems

*•* Be able to write specifications in predicate logic expressing state constraints

*•* Understand the notion of formal proof, and be able to construct simple proofs in a natural deduction proof system for predicate logic

*•* Be aware that there are inherent expressiveness and computational limitations to the applicability of logical systems, and be famil- iar with a number of restrictions under which the computational limitations can be overcome

*•* Be able to write programs in a logic programming language,

*•* Understand both the top-down and the bottom up operational semantics of logic programs

*•* Be familiar with a logic for reasoning about sequential programs, and capable of constructing correctness proofs for simple programs

(d) Teaching and Learning pattern

The course consists of a traditional lecture component and a project component. The lecture component introduces the basic concepts and techniques of logic programming. The project component will be one or several projects related to logic programming.

(e) Indicative content

*•* Introduction

*•* Pure logic (relational) programming

*•* The Prolog Language

*•* Programming in Prolog.

*•* Efficient Prolog Programming

*•* Combining Logic Programming, Functional Programming, Higher

Order, Objects.

*•* Review of first order predicate logic and resolution.

*•* Fundamental results.

*•* Semantics of logic programs.

*•* Implementation of logic languages and advanced compilation.

*•* Parallelism, concurrency.

*•* Other LP/CLP languages

(f ) Assessment method

Assessment will be by assignments and/or tests (40%) and written examination (60%)

(g) Reading list

(i) Logic in Computer Science, Modeling and Reasoning about Sys- tems, M.R. Huth and M.D. Ryan, Cambridge University Press

2000

(ii) SWI Prolog Home Page, <http://www.swi-prolog.org/>