UNIVERSITY OF NEBRASKA AT OMAHA

COURSE SYLLABUS/DESCRIPTION

Department and Course Number	CIST 1940
Course Title	Introduction to Functional Programming
Course Coordinator	Victor Winter
Total Credits	3
Date of Last Revision	February 17, 2014

1.0 Course Description

1.1 Catalog description:

This is an elective course whose purpose is to introduce math and computer science students to SML, a clean functional programming language. The topics covered provide a good background for a number of courses in the CSCI curriculum including CSCI 2030, CSCI 3660, and CSCI 4220.

1.2 Prerequisites of the course:

1.2.1 MATH 1310 (or equivalent)

1.3 Overview of content and purpose of the course

The functional programming paradigm has a clean semantics that essentially amounts to a mechanization of a restricted form of equational reasoning. This is in sharp contrast to languages like C, C++, or Java whose formal semantics are somewhat unwieldy. Studies have shown that introducing students to a functional programming language early on can be very beneficial to their development as programmers, even when their primary programming language is non-functional.

The purpose of this course is to introduce students to SML, a clean functional programming language. Programming concepts will be taught using an API called *Bricklayer* which is being developed in-house at UNO. Bricklayer provides abstractions enabling students to write SML programs whose execution produce LEGO structures. A Bricklayer website is being developed containing educational materials such as coding examples and Bricklayer documentation.

http://faculty.ist.unomaha.edu/winter/Bricklayer/index.html

The LEGO structures that can be created through Bricklayer together with the functionality provided by Bricklayers API allow for a gentle introduction to recursion.

1.4 Unusual circumstances of the course

None

2.0 Course Justification Information

2.1 Anticipated audience / demand:

This course is an elective intended for students pursuing a degree in computer science and related disciplines.

2.2 Indicate how often this course will be offered and the anticipated enrollment:

This course will be offer annually with an anticipated enrollment of 10-20 students.

2.3 If it is a significant change to an existing course, please explain why it is needed:

NA

3.0 List of performance objectives stated in learning outcomes in a student's perspective:

The student that successfully completes this course will:

1. be able to:
   a. write well-structured non-recursive programs in SML,
   b. write simple recursive programs in SML,
   c. apply basic inductive and equational reasoning techniques to develop and reason about simple recursive programs.
2. have:
   a. an increased ability in computational thinking,
   b. a rudimentary understanding of simple types,
   c. an understanding of functions as values, and
   d. a semi-formal understanding of the semantics of the core of the SML programming language.

4.0 Content and Organization

List of topics covered in chronological sequence

4.1 Notation and Terminology..3
 4.1.1 Inference rule notation (natural deduction)
4.2 Simple functional programs and their evaluation........................9
 4.2.1 primitive values and operators
 4.2.2 tuples
 4.2.3 declarations
 4.2.4 functions
 4.2.5 simple evaluation rules -- small-step operational semantics
 4.2.6 types
4.3 Induction and Recursion..9
 4.3.1 Countability
 4.3.2 Well-founded Ordering
 4.3.3 Recursive programs
4.4 Advanced topics..1.5

5.0 Teaching Methodology

5.1 Methods

The primary teaching method will be lecture, demonstration, and discussion. Examples and exercises will be primarily based on LEGO structures that can be created using Bricklayer. Given the artistic possibilities of LEGO, a successful type of assignment is an open-ended “art show”. Students can work in groups or individually to create any LEGO structure they desire and the results will be viewed by all during class. Such art shows have proven to be extremely motivating.

5.2 Student role in the course

The student is expected to attend lectures, participate in discussion on assigned readings, complete assigned homework problems, quizzes, and pass a midterm and final examination.

5.3 Contact hours:

3 hours per week

6.0 Evaluation Information

6.1 Describe the typical types of student projects that will be the basis for evaluating student performance:

Student products will be examinations, on-line quizzes as well as homework assignments consisting of hand-solved problems as well as programs written in SML.

6.2 Basis for determining the final grade:

Homework and Quizzes:	50%
Midterm:	25%
Final:	25%

6.3 Grading scale

Grades will range from A to F, with the specifics given by the table shown below. This information will be contained in the course outline and made available on the first day of class.

Grade	**Point Value**
A+	$96 \leq x \leq 100$
A	$92 \leq x < 96$
A-	$89 \leq x < 92$
B+	$86 \leq x < 89$
B	$82 \leq x < 86$
B-	$79 \leq x < 82$
C+	$76 \leq x < 79$
C	$72 \leq x < 76$
C-	$69 \leq x < 72$
D+	$66 \leq x < 69$
D	$62 \leq x < 66$
D-	$59 \leq x < 62$
F	Less than 59

7.0 Resource Material

7.1 Textbook or other required readings used in course

Ullman, J. D. (1998). *Elements of ML Programming, ML97 Edition (2nd Edition)*. Upper Saddle River, NJ: Prentice-Hall.

7.2 Other suggested reading materials, if any

Hansen, Michael & Rischel, Hans (1999). *Introduction to Programming using SML (International Computer Science Series)*. New York, NY: Addison-Wesley.

7.3 Current bibliography of resources for student's information

Paulson, L. C. (1996). *ML for the Working Programmer*. Cambridge, UK: Cambridge University Press.

Gansner, E. R. & Reppy, J. H. (2004). *The Standard ML Basis Library*. Cambridge, UK: Cambridge University Press..

Milner, R., Harper, R., MacQueen, D., & Tofte, M. (1997). *The Definition of Standard ML - Revised*. Cambridge, MA: MIT Press.

Reade, C. (1989). *Elements of Functional Programming*. New York, NY: Addison-Wesley.

Appel, A. W. (1998). *Modern Compiler Implementation in ML*. Cambridge, UK: Cambridge University Press.

Sokolowski, S. (1991). *Applicative High Order Programming - The Standard ML Perspective*. New York, NY: Chapman & Hall Computing.

Bird, R. (2010). *Pearls of Functional Algorithm Design.* Cambridge, UK: Cambridge University Press.

Petricek, T. & Skeet, J. (2009). *Real World Functional Programming: With Examples in F# and C#*. Greenwich, CT: Manning Publications.

Smith, J. B. (2006). *Practical OCaml*. New York, NY: Springer-Verlag.

Cousineau, G., Mauny, M., & Callaway, K. (1998). *The Functional Approach to Programming*. Cambridge, UK: Cambridge University Press.

8.0 Other Information:

8.1 Accommodations statement:
8.2 Other:
8.3 Author(s):

Victor Winter

9.0 Estimate Computer Science Accreditation Board (CSAB) Category Content (class time in hours):

CSAB Category	*Core*	*Advanced*
Data structures	4	0
Computer organization and architecture	0	0
Algorithms and software design	16	0
Concepts of programming languages	8	0

10.0 Oral and Written Communications:

Every student is required to submit at least 0 written reports (not including exams, tests, quizzes, or commented programs) to typically 10 pages and to make 0 oral presentations of typically 0 minutes duration. Include only material that is graded for grammar, spelling, style, and so forth, as well as for technical content, completeness, and accuracy.

11.0 Social and Ethical Issues:

No coverage

12.0 Theoretical content:

12.1 Term algebras..3
12.2 Small-step operational semantics...3
12.3 Induction and Recursion...6
- 12.3.1 Countability
- 12.3.2 Well-founded Ordering

12.4 Matching..6
12.5 Equational Reasoning...1.5
12.6 Higher-order functions...3
12.7 Rewrite systems..2
12.8 Lazy evaluation...1

13.0 Problem analysis:

Students are exposed to problem analysis and problem solving through class examples and programming assignments.

14.0 Solution design:

A sequence of programming assignments provide students with hands-on experience in the design and implementation of functional programs.

CHANGE HISTORY

Date	***Change***	***By whom***	***Comments***
09/07/2007	Initial proposal	Winter	Submitted to CSCI and Math
09/14/2011	Resubmission of proposal	Winter	Submitted to CSCI
09/22/2011	Revision and cleanup	Winter	
10/08/2011	References converted to APA	Winter	
02/17/2014	Revised to reflect change in emphasis	Winter	