# Functional and Logic Programming

**Prerequisites:** A grade of C$^-$ or better in CPSC 141, and CPSC 281; or permission of instructor.

**Accommodations:** If there is any student in this course who, because of a disability, may have a need for special academic accommodations, please come and discuss this with me, or contact the Access Resource Centre located in Teaching & Learning 10-1048.

**Web-page:** `http://casper.unbc.ca/Semesters/2015F/370.php`

**Instructor:** David Casperson; **Office:** T&L 10-2040; **Phone:** 960-6672; **Departmental Assistant:** Marva Byfield (T&L 10-2074 960-6490); **e-mail:** `David.Casperson@unbc.ca` .

**Lecture times:** M W F 13:30–14:20. **Room**: 5-159. There are *no* assigned lab or tutorial times.

**Office Hours:** Scheduled for:
Mon 11:00-12:00
Mon 14:30-16:00.

**Text Books:** None are required. [5] is a great introduction to HASKELL conveniently found on the web. [6, 1] are both in the library.

**Grading Scheme:**

| | | |
|---|---|---|
| Homework: | 25% | |
| Midterm 1: | 20% | Wed, Oct 14 |
| Midterm 2: | 20% | Mon, Nov 09 |
| Final Exam: | 35% | 3h in 2015-12-08 *to* 2015-12-18 |

*I reserve the right to change the weight of any portion of this marking scheme.* If changes are made, your grade will be calculated using the original weighting and the new weighting, and you will be given the higher of the two.

**Programming Assigments:** There will be approximately weekly programming assignments during the semester. Programming languages include SCHEME, PROLOG and HASKELL.

**Approximate Course Content:**

GENERAL
- An introduction to functional programming and languages.
- Language concepts: Static and dynamic typing. Strict and non-strict evaluation. "Pure" versus "impure" languages.
- Mathematical concepts: Relations, Functions, partial functions, Cartesian products, Disjoint unions, "Currying".
- Programming: recursion, tail recursion, tail recursion strategies.

HASKELL:
- Basic syntax.
- Types, polymorphic types, type classes
- Creating data structures.
- Exploiting laziness. Downsides to laziness.
- Monads, `do`-notation, monadic programming.

SCHEME:
- Basic syntax.
- Space and time complexity for functional programs and data structures.
- the pure $\lambda$-calculus.
- Combinators.

PROLOG
- *An introduction to logic programming.*
- Facts. Rules. Goals. Variables. Conjunctions. Horn Clauses.
- The Unification algorithm.
- Programming strategies: Accumulator arguments, difference lists.
- Cuts. Negation.
- Arithmetic.
- Debugging.

## References

[1] W. F. Clocksin and C. S. Mellish, *Programming in Prolog*, Springer Verlag, 1981.

[2] Matthias Felleisen and Dan Friedman, *The little MLer*, MIT Press, 1998, in the UNBC library.

[3] Daniel P. Friedman and Matthias Felleisen, *The little Schemer*, fourth ed., The MIT Press, 1996.

[4] ———, *The seasoned Schemer*, The MIT Press, 1996.

[5] Miran Lipovaca, *Learn You a Haskell for Great Good!: A Beginner's Guide*, 1st ed., No Starch Press, San Francisco, CA, USA, 2011.

[6] Richard A. O'Keefe, *The Craft of Prolog*, Logic Programming, MIT Press, 1990.

[7] Larry C. Paulson, *ML for the working programmer*, second ed., Cambridge University Press, 1996, This is more in-depth than Ullman's book.

[8] Riccardo Pucella, *Notes on programming in in Standard ML of New Jersey*, Cornell University, January 2001, also available from `http://www.smlnj.org//doc/literature.html\#tutorials`.

[9] P. L. Wadler, *Comprehending monads*, Proceedings of the 1990 ACM Conference on LISP and Functional Programming, Nice (New York, NY), ACM, 1990, pp. 61–78.