**COS 498 Parallel and Concurrent Programming in Haskell**

# OVERVIEW

Haskell is a high-level, pure functional programming language with a strong static type system and elegant mathematical underpinnings, and is being increasingly used in industry.

60-70% of the course will be devoted to exploring the joys of pure, lazy, typed functional programming in Haskell. The materials will include Brent Yorgey’s CIS 194 [course](http://www.seas.upenn.edu/~cis194/spring13/lectures.html)[[1]](#footnote-1), NICTA Functional Programming [course](https://github.com/NICTA/course)[[2]](#footnote-2) and two books on Haskell: *Real World Haskell* by Bryan O'Sullivan, Don Stewart, and John Goerzen and *Learn You a Haskell for Great Good* by Miran Lipovača.

The remaining time will be devoted to exploring parallel (including GPU-based) and distributed computing using Haskell. The primary book on the topic would be *Parallel and Concurrent Programming in Haskell* by Simon Marlow.

# COURSE OBJECTIVES

* Learn pure functional programming
* Gain practical experience of solving problems functionally using Haskell
* Explore parallel and concurrent programming

# EVALUATION

Both CIS 194 and NICTA courses include programming assignments, which will be used to evaluate student’s performance during the first part of the course. An open-ended final project will conclude the second part.

Homeworks - CIS 194 35%  
Exercises - NICTA course 40%  
Final project 25%

# 

# 

# GRADING

* 96 - 100% = A (4.00)
* 90 - 95% = A- (3.67)
* 87 - 89% = B+ (3.33)
* 83 - 86% = B (3.00)
* 80 - 82% = B- (2.67)
* 77 - 79% = C+ (2.33)
* 73 - 76% = C (2.00)
* 70 - 72% = C- (1.67)
* 67 - 69% = D+ (1.33)
* 60 - 66% = D (1.00)
* 59% and below = F

# BREAKDOWN

## Weeks 1-5 - CIS 194

Introduction to Haskell  
Algebraic Data Types  
Recursion patterns, polymorphism and the Prelude  
Higher-order programming and type inference  
More polymorphism and type classes  
Lazy evaluation  
Folds and monoids  
IO  
Functors  
Applicative functors  
Monads

## Weeks 6-10 - NICTA

The course is structured as a linear progression, explaining more advanced concepts related to functional programming in Haskell. The following modules will be covered: Id, Optional, Validation, Functor, Applicative, Monad, FileIO, State, StateT, Extend, Comonad, Compose, Traversable, ListZipper, Parser, MoreParser, JsonParser, Interactive, Anagrams, FastAnagrams, Cheque. Due to time constraints, not all of the exercises might be completed.

## Weeks 11-14

Research: libraries and packages for parallel/GPU-based/distributed computing, algorithms and/or MapReduce. Writing a final project.

# ACADEMIC HONESTY

The University policies regarding academic honesty will be followed and enforced, if needed.

1. http://www.seas.upenn.edu/~cis194/spring13/lectures.html [↑](#footnote-ref-1)
2. https://github.com/NICTA/course [↑](#footnote-ref-2)