# 2022-Fall

**COMP 311 / COMP 544: Functional Programming (Fall 2022)**

| Syllabus | Online Book | Racket HW Guide | Racket HW Grading | Java HW Guide | SVN Documentation | HW Support Documents |
|---|---|---|---|---|---|---|

| | | | |
|---|---|---|---|
| **Instructor** | Robert "Corky" Cartwright | | |
| **Lectures** | DCH 1064 | **Lecture Times** | 9:25am–10:40am TR |
| **Instructor Email** | cork@rice.edu | **Online Discussion** | Piazza – Rice Comp 311 |

## Brief Description

This class provides an introduction to functional programming. Functional programming is a style of programming in which computations are solely expressed in terms of applications of functions to arguments (which themselves can be functions). This style of programming has a long history in computer science, beginning with the formulation of the Lambda Calculus as a foundation for mathematics. It has become increasingly popular in recent years because it offers important advantages in designing, maintaining, and reasoning about programs in modern contexts such as web services, parallel (multicore) programming, and distributed computing. Course work consists of a series of programming assignments in the Racket, Java, and Haskell programming languages plus occasional written homework assignments on underlying theory.

**Grading, Honor Code Policy, Processes, and Procedures**
Grading will be based on your performance on weekly programming assignments and two exams: a midterm and a final. All work in this class is expected to be your own, and you are expected not to post your solutions or share your work with other students, even after you have taken the course. Please read the Comp 311 Honor Code Policy for more details on how you are expected to work on your assignments. There will also be a final exam, as described in the syllabus.

All students will be held to the standards of the Rice Honor Code, a code that you pledged to honor when you matriculated at this institution. If you are unfamiliar with the details of this code and how it is administered, you should consult the Honor System Handbook. This handbook outlines the University's expectations for the integrity of your academic work, the procedures for resolving alleged violations of those expectations, and the rights and responsibilities of students and faculty members throughout the process.

**Accommodations for Students with Special Needs**

Students with disabilities are encouraged to contact me during the first two weeks of class regarding special needs. Students with disabilities should also contact Disabled Student Services in the Ley Student Center and the Rice Disability Support Services.

## General Information

**Office Hours**

| **Instructor** | | | |
|---|---|---|---|
| | | | |
| Corky Cartwright | TuTh | 3pm-4pm | DCH 3104 |
| | | By appointment | TBA |
| **Teaching Assistants** | | | |
| Clayton Ramsey | TBA | TBA | TBA |
| Andrew Obler | TBA | TBA | TBA |

**Textbooks**

There is no required textbook. We will follow the pedagogic approach of "How to Design Programs, First Edition" and extend it to other languages. The Second Edition of this book is default at the website www.htdp.org but this page contains a link to the first edition (at URL: https://htdp.org/2003-09-26/) at the bottom of the page. We will draw material from a variety of sources, including:

- Felleisen, Findler, Flatt, Krishnamurthi. "How to Design Programs, First Edition" MIT Press 2001.
- Robert Cartwright, "The Elements of Object-Oriented Design", Unpublished notes.
- Harold Abelson, Gerald Jay Sussman, Julie Sussman, "The Structure and Interpretation of Computer Programs." MIT Press 1985.
- Odersky, Spoon, Venners. "Programming in Scala." Artima Press 2012.
- Chiusano and Bjarnason. "Functional Programming in Scala." Manning Publications Co. August 2014.
- Coursera: Functional Programming Principles in Scala by Martin Odersky.
- edX: FP101x: Introduction to Functional Programming by Erik Meijer.
- Okasaki. "Purely Functional Data Structures." Cambridge University Press. New York, NY. 1999.
- Cartwright, "Why is Functional Programming Important?" (Advanced material)

**Recommended Videos**

- *Working Hard to Keep it Simple*, by Martin Odersky
- *Growing a Language*, by Guy L. Steele, Jr.
- *What to Leave Implicit,* by Martin Odersky

| | |
|---|---|
| Development Environment | • DrRacket is recommended for all Racket homework assignments in this course. The interface is "textually transparent" as we will show in class.<br>• DrJava is the supported IDE for Java in this course, but you are welcome to use any IDE such as IntellJ or Eclipse.<br>• We are still evaluating IDEs for Rust. |

## Lecture Schedule (In Progress)

| Week | Day | Date | Lecture Topic and Resources | Work Assigned | Work Due |
|---|---|---|---|---|---|
| 1 | Tu | Aug 23 | Motivation and the Elements (Constants) of Racket | Skim HTDP First Edition, Part 1 (Ch 1-8), Part 2 (Ch 9-10) | Sep 01 |
| | Th | Aug 25 | Conditionals, Function Definitions and Computation by Reduction | | Sep 03 |
| 2 | Tu | Sep 01 | Conditionals, Function Definitions and Computation by Reduction | Homework 1<br>Review Ch 8<br>HTDP Part 2 (Ch 9-10) | Sep 08 |
| 3 | Th | Sep 03 | The Program Design Recipe for Racket focusing on using recursion<br>to process lists and natural numbers | Preface, 9.4<br>HTDP Part 2 (Ch 11-13) | Sep 10 |
| 4 | Tu | Sep 06 | Data Definitions, Data-driven Structural Recursion, | Homework 2<br>HTDP Part 3 | Sep 15 |
| 5 | Th | Sep 08 | Mutually Recursive Definitions and Help Functions | HTDP Ch 15-17 | Sep 15 |
| 6 | Tu | Sep 13 | Local Definitions and Lexical Scope | Homework 3<br>HTDP Parts 5-6 | Sep 22 |
| 7 | Th | Sep 15 | Lambda the Ultimate and Reduction Semantics | LawsOfEvaluation | Sep 22 |
| 8 | Tu | Sep 20 | Functional Abstraction and Polymorphism | | Sep 29 |
| 9 | Th | Sep 22 | Functions as Values | Homework 4 | Sep 29 |
| 10 | Tu | Sep 27 | Generative (Non-structural) Recursion | | |
| 11 | Th | Sep 29 | Lazy Evaluation and Non-strict Constructors | Homework 5* | Oct 11 |
| 12 | Tu | Oct 04 | Techniques for Implementing Lazy Evaluation | | |
| 13 | Th | Oct 06 | A Glimpse at Imperative Racket and Memoization | Sample Exam | |
| | Tu | Oct 11 | Fall Recess | Sample Exam Key | |
| 13 | Th | Oct 13 | On to Java! | OO Design Notes | |
| 14 | Tu | Oct 18 | Adapting the HTDP Design Recipe to Java<br>Midterm (Through Lecture 13 and HW 5) 7-10pm | Homework 6 | Oct 25 |
| 15 | Th | Oct 20 | Higher-order Functional Programming in Java | | Oct 31 |
| 16 | Tu | Oct 25 | Four Key Idioms for Encoding FP in Java | Homework 7 | Nov 1 |
| 17 | Th | Oct 27 | The Singleton and Visitor Patterns | | |
| 18 | Tu | Nov 01 | Java Generics and Their Role in FP in Java | Homework 8* | Nov 10 |
| 19 | Th | Nov 03 | Functional Rust I | | |
| 20 | Tu | Nov 08 | Functional Rust II | | Nov 15 |
| 21 | Th | Nov 10 | OO Rust Using Only Traits | Homework 9 | |
| 22 | Tu | Nov 15 | Rust ... | | |
| 23 | Th | Nov 17 | Rust ... | Homework 10 | Nov 29 |
| 24 | Tu | Nov 22 | Rust ... | Homework 11* | Dec 2 |
| 25 | Tu | Nov 29 | Rust Concurrency | | |

| | | | | | |
|---|---|---|---|---|---|
| 26 | Th | Dec 1 | Future of FP | | |

*Assignments marked with * are double assignments that count twice as much as regular assignments. **indicates the project in lieu of a final examination.