Legacy

COMP 311 / COMP 544: Functional Programming (Fall 2020)

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

Instructors	Robert "Corky" Cartwright	**TA**	TBA
Lectures	Online using Zoom	**Lecture Times**	9:40am–11:00am 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	1:30pm-2:30pm	Online via Zoom
		By appointment	Online via Zoom
Teaching Assistants			
Agnishom	TBA	TBA	Online via Zoom
Andrew Obler	TBA	TBA	Online via Zoom
Chunxiao Liao	TBA	TBA	Online via Zoom

Textbooks	There is no required textbook. We will follow the pedagogic approach of "How to Design Programs" and extend it to other languages. We will also draw material from a variety of sources, including: • Felleisen, Findler, Flatt, Krishnamurthi. "How to Design Programs." 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. • "Why is functional programming important?" by Corky Cartwright
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. • DrJava is the supported IDE for Java in this course, but you are welcome to use any IDE such as IntelliJ or Eclipse. • We are still evaluating IDEs for Haskell.

Lecture Schedule (In Progress)

Week	Day	Date	Lecture Topic and Resources	Work Assigned	Work Due
1	Tu	Aug 27	Motivation and the Elements (Constants) of Racket	HTDP Part 1 (Ch 1-8)	Sep 03
	Th	Aug 29	[Canceled for Hurricane Laura]		Sep 05
2	Tu	Sep 01	Conditionals, Function Definitions and Computation by Reduction	Homework 1 Review Ch 8 HTDP Part 2 (Ch 9-10)	Sep 08
3	Th	Sep 03	The Program Design Recipe for Racket focusing on using recursion to process lists and natural numbers	Preface, 9.4 HTDP Part 2 (Ch 11-13)	Sep 10
4	Tu	Sep 08	Data Definitions, Data-driven Structural Recursion,	Homework 2 HTDP Part 3	Sep 16
5	Th	Sep 10	Mutually Recursive Definitions and Help Functions	HTDP Ch 15-17	Sep 17
6	Tu	Sep 15	Local Definitions and Lexical Scope	Homework 3 HTDP Parts 5-6	Sep 23
7	Th	Sep 17	Lambda the Ultimate and Reduction Semantics	LawsOfEval.pdf	Sep 26
8	Tu	Sep 19	Functional Abstraction and Polymorphism		Oct 6
9	Th	Sep 24	Functions as Values	Homework 4	Oct 5
10	Tu	Sep 29	Generative (Non-structural) Recursion		Oct 6
11	Th	Oct 01	Lazy Evaluation and Non-strict Constructors		
12	Tu	Oct 06	Techniques for Implementing Lazy Evaluation	Homework 5*	Oct 14
13	Th	Oct 08	A Glimpse at Imperative Racket and Memoization		
	Tu	Oct 13	Racket Review	Sample Exam Sample Exam Key	
13	Th	Oct 15	On to Java!	OO Design Notes	
	Fri	Oct 16	Midterm		
14	Tu	Oct 22	Adapting the HTDP Design Recipe to Java	Homework 6	Oct 27
15	Th	Oct 24	Higher-order Functional Programming in Java		
16	Tu	Oct 29	Four Key Idioms for Encoding FP in Java	Homework 7	Nov 6

17	Th	Oct 31	The Singleton and Visitor Patterns		
18	Tu	Nov 03	Java Generics and Their Role in FP in Java	Homework 8*	Nov 16
19	Th	Nov 05	The Strategy Pattern: Functions as Arguments in Java		
20	Tu	Nov 10	Core Haskell (call-by-name, lazy constructors) (Agnishom)		
21	Th	Nov 12	Haskell Pattern Matching (Agnishom)	Homework 9	Nov 19
22	Tu	Nov 17	Haskell Type Classes (Agnishom)		Dec 16
23	Th	Nov 19	Haskell Monads (Agnishom)	Final Project**	

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