Start Here!

Thank you for signing up for Programming Languages, Part B on Coursera. We are very glad you are here!

First things first, let's be clear that this is Part B of a 3-part course. We will assume you have completed Part A and are eager to continue our challenging and rewarding study of Programming Languages. There is not really a good way to describe the necessary background for Part B other than to say that it builds on the material in Part A in many ways. The introductory material in Part A discussed why we created a 3-part course. We don't repeat that discussion here, but you may wish to (re-)visit it.

Because the format of Part B is similar to Part A, we will not have an "introductory week" like we did in Part A -- there wouldn't be very much to do. You do need to install the software for programming in Racket, but this will not take very long.

Part B has three "weeks":

- Week 1 has a lot of content. First, we will cover various Racket basics and (quickly) see how we can program in a similar functional style to ML but with a very different syntax. Then we will focus on several programming idioms related to "delaying evaluation" by putting a computation in the body of a zero-argument function. The homework assignment is a collection of fairly short but often challenging functions on different topics.
- Week 2 first presents Racket *structs* and how they serve many of the roles of ML's datatype bindings but in a dynamically typed language. Then we will focus on the main topic of Homework 5: writing an interpreter to implement a programming language, particularly implementing function closures. While there are many fewer lectures than Week 1 and in weeks of Part A, the homework is more challenging and more rewarding.
- Week 3 leverages our experience programming in both ML and Racket, discussing what it means to have a static type system in a language and the various advantages and disadvantages of doing so. Because this material does not lend itself to programming problems, this week will just have a short quiz covering some of the content. But rest assured that the content is not "theoretical" -- it is essential perspective on how to think about type systems (or the lack thereof) to be a more effective software developer.

Note that you may find Weeks 2 and 3 shorter than usual because, in fact, they used to be part of the same week. Since the topics are nicely separable, we have "spread them out" but, of course, you can finish in less than (or more than) 3 weeks if you prefer.

There is no exam in Part B other than the Week 3 quiz. Instead, the exam at the end of Part C covers material in Part B and Part C.

The introductory videos in this first "lesson" provide a little more detail welcoming you to "Part B", discussing the topics ahead, and describing the structure of the course.

[As a final detail, in Part B and Part C, each section (e.g., 5) has a homework number that is less than the section number (e.g., 4). This is because Section 4 of Part A and Section 7 of Part B do not have homeworks. We know this is a little confusing, but skipping a homework number is probably at least as confusing to some people.]