too much agency for the reader, and might put them on the right track as they consider this challenge.

Chapter 13 introduces networking for communication between game instances, and Chapter 14 brings it all together in one final game, 'Hungry Henry'. Finally, with a 'close paren', *Realm of Racket* leaves you with hints of where you might go: object orientation, syntactic meta-programming, and the design of languages and their interpretation.

Conclusions

Realm of Racket is a book written 'by freshmen, for freshmen'. It introduces readers with some programming experience to Racket, a language in the Scheme family. This introduction leverages the years of extensive work that have gone into Racket (the language) and DrRacket (the program development environment), and through the development of games, prepares the reader for thinking seriously about the design of complex, interactive programs. The design of structured, linked data (game trees), lazy evaluation, memorization, and the basics of search and AI await the reader who makes their way through to the end of the text. It isn't quite a textbook, and it certainly isn't 'Game Programming in 21 Days'. It is, however, a wonderful text for delving into a language with a long and rich history.

The text is informal, the illustrations whimsical and fun, and flamethrower-wielding squirrels await the intrepid programmer who makes their way through to the end. Indeed, I wonder if I might have picked up Scheme more easily, back in the day, had a text like this existed for me at the time. It puts important ideas (both regarding the language and more generally regarding interactive application design) in a context that is natural, and it does it well. All of that said, I am easily swayed by things like rodents armed with military-grade weapons; but, when it comes down to it, I can't help but ask: what isn't there to like?

MATT JADUD
Computer Science, Berea College, Berea, KY, USA

Beginning Haskell, by Alejandro Serrano Mena, Apress, New York City, NY, 2014, ISBN-10: 1430262508, 428 pp.
doi:10.1017/S0956796814000112

This book is a modern introduction to Haskell. It is refreshingly up-to-date, covering topics such as the EclipseFP plugin for the Eclipse IDE, view patterns, and even – towards the end of the book, as a taster for going beyond Haskell – dependently typed programming in Idris. (It is worth noting that recent Haskell language extensions available in the Glasgow Haskell Compiler (GHC) do provide alternatives to full-fledged dependent types, so even this chapter could be indirectly useful to a professional Haskell programmer.)

The book assumes, very reasonably, that the reader will be using a recent version of GHC, and that the Haskell 2010 standard (modulo a few tweaks) will constitute the default language environment to work in. Although Haskell programs are usually compiled, the book sensibly starts by introducing the reader to the Haskell interpreter GHCi – an invaluable tool for learning Haskell and for quickly checking one's understanding of a concept or function. The book also introduces the reader to EclipseFP, but it is always presented as a convenient alternative, not as a tool that must be used, which I think is the right approach.

Topics are introduced in a logical order – but that does not mean according to a dry, dusty, strict taxonomy. Interesting asides and useful tips are inserted at relevant points in the text. Priority is given to clear step-by-step exposition over immediately introducing an

idiomatic style of Haskell. However, the pace is quick – this is not a book for those who are new to programming, as the Introduction notes. Programmers who are coming to Haskell from languages other than C or C++ may find themselves shaking their heads at some minor mistakes (for example, the book assumes that all languages other than Haskell have 'C-style' arrays, which is not true of, say, Java).

A humorous running example of a time machine shop is used throughout the book. The fact that this running example is obviously fictitious might at least shield the book from a certain amount of pedantic objections from experienced programmers who might otherwise seize on an example and say 'that is too simplistic – I have implemented that, and in reality it was more complicated'. Of course, that would be to miss the point. As one would expect from this type of book, there was clearly an attempt to construct examples with an eye to clarity and concision, rather than commercial realism.

Unfortunately, this aim of clarity and concision is rather let down by persistent sloppiness. The book is riddled with trivial mistakes: at least three cases in which the code – while it would compile – does not quite match the prose that purports to describe it; numerous grammatical errors; and even some spelling mistakes in the prose, which a spellchecker certainly could have caught. Even the very first piece of Haskell code in the book is wrong – the arguments to the map function are the wrong way round (and arguably 'action' is not a very good name to represent a pure function, either). While, as a native English speaker and an experienced Haskell programmer, I could easily glide over these mistakes and discern the intended meaning, for someone new to Haskell or a non-native speaker of English, these could be quite confusing and an impediment to learning.

Moreover, this is not the first programming book I have noticed recently to have suffered from a glaring lack of competent copy-editing on the part of its publisher. I do not remember this being a problem in the past. It is not necessarily a matter of technical awareness: one does not need to know anything about programming to notice that the phrase 'a special comment that is interpreter by the compiler' is ungrammatical. I do not know what more to say about this problem, except to implore both readers and authors to vote with their feet and avoid publishers that do not invest in careful copyeditors. However, Haskell books are rather thin on the ground at the best of times, so perhaps that suggestion is impractical in this case.

ROBIN GREEN
London, UK
(e-mail: greenrd@greenrd.org)

https://doi.org/10.1017/S0956796814000112 Published online by Cambridge University Press