Preface

It has long been recognized that there is a relationship between logic and computer programming. Since the 1940s many logicians, including Alan Turing and John von Neumann, and computer scientists such as Edsgar Dijkstra and John McCarthy, have drawn attention to the connection between the two disciplines and commented on its importance. During the formal methods boom of the 1980s and 1990s it was common to read assertions to the effect that programming could be reduced to a quasi-logical process of theorem proving and formula derivation.

However, despite having a background in both programming and logic, I found something elusive about such assertions. My practical experience of programming had led me to understand it as a very different type of activity from the formal manipulations of logical proof. With the intention of resolving this uncertainty, I embarked upon a Ph.D. to study the ways in which the development of programming languages had been influenced by logic. This book is based on the resulting thesis, although it has a wider focus and marks a further stage in my growing understanding of the connection between the two areas.

Broadly speaking, I have come to understand this connection not as a *fact* about the two disciplines, but as something more like a *decision* made by identifiable historical actors as to how the new discipline of programming should be understood and structured. A key event in this process was the creation of the Algol language in the years around 1960. The motivation for this decision has very deep roots, however, reaching back to ideas about machines and machinic processes that date from the very beginning of the scientific revolution.

From this starting point, the book gives an account of the history of what we now call programming. It is useful, however, to view this not simply as *computer* programming, but to think more generally of attempts to define the steps involved in computations and other information-processing activities in such a way that they could be performed by machines, or at least by humans mimicking the behaviour of machines. From this perspective, the history of programming is distinct from the history of the computer, despite the close relationship between the two in the twentieth century.

The story told in this book stops in the 1970s, at the point where object-oriented programming emerged as a successor, or alternative, to structured programming. This is not an arbitrary cut-off point; rather, I believe that structured programming marks the completion of a particular historical trajectory, and that object-orientation, despite its rich inheritance from existing practices, is something quite different, or at least something which has deep roots in approaches and disciplines other than logic. Untangling these roots is part of a different project, however, and the history of how programming and programming languages developed after the 1970s is part of a different story from the one told here.

There are several things this book is not, therefore. Although it contains much historical material, it does not pretend to be a complete history of programming languages, and still less a history of the computer. What it does try to do is to tell a particular story about these historical developments and to show one way in which the history of programming languages can be set in a wider context and seen as an integral part of a development which, ultimately, is central to the project started by the scientific revolution in the seventeenth century.

The history of computing has recently been expanding its focus from technical details and embracing wider narratives and historical contexts. These encouraging developments have been principally evident in studies of commercial and military applications of computing and their societal impacts. If I had a single hope for this book, it would be that it might contribute to a similar process in the more technical aspects of the subject and to inspire computer scientists and historians of science and technology to see programming not as a isolated technical field. but as an interesting and important part of general intellectual history.

Acknowledgements I would like particularly to thank my supervisor Donald Gillies and external examiner John Tucker, without whose enthusiastic advice, support and encouragement this book would never have seen the light of day.

London, UK Mark Priestley

http://www.springer.com/978-1-84882-554-3

A Science of Operations
Machines, Logic and the Invention of Programming
Priestley, M.
2011, X, 342 p., Hardcover
ISBN: 978-1-84882-554-3