

---

## Preface

Computer programming is one of fundamental areas in engineering. As computers have permeated our modern lives, it has been increasingly more attractive to write programs to make these machines work for us. Only a couple of decades ago, a computer course was the first time that a student met with a computer. Today, a standard first-year undergraduate student has at least ten years of experience on using programs and diverse software on their desktops, laptops, and smart phones. But, interestingly, when it comes to writing programs in addition to using them, programming courses and materials considered in those mandatory practical hours remain as “difficult stuff” for many students, who are even experts in using their technological gadgets.

There are extremely many books in computer programming, some of which are excellent sources for teaching and learning programming and related concepts. Programming would be incomplete without explaining underlying algorithms. Hence, most of these books also cover algorithmic techniques for solving problems, which are usually accompanied by some coding techniques using a programming language or pseudocodes. I am also using various books in my own courses. Some of them contain hundreds of pages with nice discussions on programming and algorithms. On the other hand, I have witnessed that, when they have trouble to understand a concept or a part of material, many students prefer internet, such as discussion boards, rather than their books. Their responses to my question, i.e., why they are not willing to follow their books, has forced me to write this one, not to replace other texts in this area, but to support them via an introductory material that many student find quite easy to follow.

My discussions with students have often led to the same point that they admit what they find difficult while programming. I have found interesting that students are actually very successful to understand some critical concepts, such as recursion, that many lecturers and instructors consider difficult. On the other hand, they are struggling on implementing algorithms and writing their own programs because of some common mistakes. These “silly” mistakes, as called by students themselves, are not written in books, and they are difficult to solve since programming environments do not provide sufficient feedback on their mistakes. My reaction has been collecting these common mistakes and including them in course materials that have significantly boosted student performance. This book also contains such faulty programs written by students along with discussions for better programming.

When it comes to the point where I need to tell what is special about this book, I would describe it as a simple, concise, and short material that may be suitable for introductory programming courses. Some of the discussions in the text may be found as “stating the obvious” by many lecturers and instructors, but in fact, I have collected them from my discussions with students, and they actually include answers to those questions that students are often embarrassed to ask. I have also filtered topics such that only threshold concepts, which are major barriers in learning computer programming, are considered in this book. I believe that higher-level topics can easily be understood once the topics focused in this book are covered.

This book contains nine chapters. The first chapter is an introduction, where we start with simple examples to understand programming and algorithms. The second and third chapters present two important concepts of programming, namely loops and recursions. We consider various example programs, including those with mistakes that are commonly experienced by beginners. In the fourth chapter, we focus on the efficiency of programs and algorithms. This topic is unfortunately omitted or skipped fast in some programming courses and books, but in fact, it is required to understand why we are programming. Another important aspect, i.e., accuracy, is focused in the fifth chapter. A major topic in computer programming, namely, sorting is discussed in the sixth chapter, followed by the seventh chapter that is devoted to linear systems of equations. In the eighth chapter, we briefly discuss file processing, i.e., investigating and modifying simple files. Finally, the last chapter presents some mini projects that students may enjoy while programming.

As the title of this book suggests, all programs given in this book are written in the R language. This is merely a choice, which is supported by some of its favorable properties, such as being freely available and easy to use. Even though a single language is used throughout the book, no strict assumptions have been made so that all discussions are also valid for other programming languages. Except the last one, each chapter ends with a set of exercises that needs to be completed for fully understanding the given topics because programming and algorithms cannot be learned without evaluating, questioning, and discussing the material in an active manner via hands-on practices.

Enjoy it!

Ankara, Turkey

Özgür Ergül



<http://www.springer.com/978-1-4471-5327-6>

Guide to Programming and Algorithms Using R

Ergül, Ö.

2013, XI, 182 p., Hardcover

ISBN: 978-1-4471-5327-6