JEAN ETTINGER
Programming in C++. Macmillan. 1994. ISBN 0-333-60682-5. £14.99. 157 pp. softbound.

This is yet another introductory book on C++. What makes it different from others, is its conciseness and directness of approach. It sets out to explain the features of the C++ language and it does precisely that using simple programming examples. For a language that is as complex as C++, it is creditable that the author could present all the major features in so few pages. The deficiency of the book also lies in this: the nuances and subtle semantics that make the language a difficult one to master are never indicated. This may be justified on the grounds that it is an introductory book and ought not to scare a beginner away from the language. However, there is also the danger that a reader ends up with over-simplified views of the language. A major omission of the book, which needs to be pointed out in this context, is a list of references for further reading.

The book is structured linearly. Starting with a brief history of the language, the book describes in the first five chapters features of the language that support traditional procedural programing. These are data types and operators, derived types, control structures and functions, i.e. mostly C language features. The major C++ features are covered in the next seven chapters. The topics include file input/output, classes and class inheritance, operator and function overloading, dynamic memory management, template functions and classes, and virtual functions and virtual base classes. There is some useful material in the appendices for those who have acc[illegible] either GNU's g++ or Borland/Turbo C++ c[illegible]

Eac[illegible] brief and each point is well illustrated with [illegible]d complete programming example. Withi[illegible] structure, the author has successfully avoid[illegible]ferencing except in a few places, such as in [illegible] on page 44 and on page 62 where conce[illegible] member function and virtual derived class [illegible] in later chapters) are used. The progra[illegible]xercises given at the end of each chapter, though f[illegible] are good.

The book contains a few inaccuracies, often subtle. Some of them can be considered harmless, such as the statement that indirect selection is a binary operator. While i[illegible] may appear to be a binary operator for a casual reader, the C++ reference manual clearly treats this operator as a unary one for the purpose of overloading. The inaccuracy on p. 74, while discussing destructor member functions, is more serious. Section 7.1.4. is clearly misleading and suggests that destructor member functions are functions by which objects are destroyed. The author uses the word *destructed* and italicizes it. Yet it does not stop a reader from interpreting it in the above fashion. In fact, the sentence that follows the statement reinforces this wrong belief. The section fails to convey the fact that destructor is the member function of a class that get automatically invoked when an object of that class is deleted and typically contains code to perform actions complementary to that of the constructor.

Support for object oriented programming is the major goal of the C++ language. Therefore, in a book with a title like such as *Programming in C++* one would expect to see more about object-oriented programming than appears in this book. The grouping of virtual functions and virtual base classes into a single chapter is rather unusual since they have very little in common. A virtual function is a simpler concept to understand and is a more commonly used feature in object-oriented programming than virtual base classes.

The book will serve as a good introductory text for C++, particularly for those students who have little prior knowledge of C programming. And like many introductory level books, this book also lacks contents with any significant reference value.

T. M. VIJAYARAMAN
National Centre for Software Technology
Bombay, India

MITCHELL L. MODEL
Data Structures, Data Abstraction: A Contemporary Introduction using C++. Prentice-Hall International. 1994. ISBN 0-13-291279-1. £21.10. 502 pp. softbound.

This book serves two purposes—first, it introduces a number of data structures that are commonly used by computer scientists; and second, it introduces programming using the data abstraction and the data encapsulation features of the C++ language. In fact, it does both these jobs remarkably well.

The data structures covered in this book are the ones normally found in any data structure book. These include stacks, lists, queues, trees, graphs and tables. For every structure, a complete implementation is provided in C++ along with a small sample of application code using the data structure. The program code is made available on a floppy so that a student can play around with it using any of the common C++ compilers. The other positive aspects of this book are the following: generic implementations of the structures using the template feature of C++, a well structured presentation, a number of interesting exercises at the end of every chapter and a highly focussed introduction to C++. Although the book is not about object-oriented programming, the appendices cover the salient points of object-oriented programming and C++.

On the negative side, the book leaves applications of many of the data structures as exercises. This is particularly true of the data structures described in the later parts of the book. Also, the book ignores file-based data structures completely. The author states that this topic should not be part of a data structure book and should be relegated to advanced courses dealing specifically with file management and database implementation.

What is required is not details about specific structures like indexed sequential files, but a discussion of what makes external data structures very different from internal data structures. Since the in-core- B-tree is presented in the book, this addition could have been done with very little effort.

On the whole, it is an interesting book since it uses C++, a language that can express data abstraction better than the conventional languages, in presenting the implementation of data structures. The emphasis on the structural aspects rather than the algorithmic aspects is a welcome change, since this emphasis is the very essence of programming in languages like C++. It is ideally suited for an introductory course on data structures for students with some knowledge of C.

T. M. Vijayaraman
National Centre for Software Technology
Bombay, India