C Programming: A Modern Approach
Author: K. N. King
Edition: Second
Publisher: W. W. Norton & Company
Pages: 832
ISBN 9780393937950

C is now a rather venerable programming language. It was designed at Bell Labs in the early 1970s as a response to the problems of coding Unix, thus being developed as a low-level assembler language. Thompson regarded it as a high-level, structured programming language, and it is excellent in this role. I am working on a large industrial project that uses C programming for the parts of the system that are intimately involved with its low-level interfaces. So C is alive and well despite successors such as C++, C# and Java, which impose and encourage a stricter discipline on programming style, often with an object-oriented flavour. Indeed, early versions of C compilers, designed in the early 1980s by Bjarne Stroustrup, compiled to C as the “low-level” target language, which could then itself be compiled to object code.

The first edition of this text, from 1996, covered the C89 standard of 1989; this second edition has been updated to follow the C99 standard of 1999, with differences clearly indicated. The text has also been thoroughly updated to take references to older operating systems such as DOS.

The volume is comprehensively, with more than 800 pages of material and a good index. Chapters include exercises, with online resources for instructors, facilitating its use as a textbook, as well as a work of reference. Appendices offer comparisons of the C89 and C99 standard “Kernighan & Ritchie” versions of the C language, together with a list of C’s standard library functions and other information. Students who use C after taking this course in the past will find this book useful as a source of help when writing C code.

The first edition of this text, from 1996, covered the C89 standard of 1989; this second edition has been updated to follow the C99 standard of 1999, with differences clearly indicated. The text has also been thoroughly updated to take references to older operating systems such as DOS.

The volume is comprehensively, with more than 800 pages of material and a good index. Chapters include exercises, with online resources for instructors, facilitating its use as a textbook, as well as a work of reference. Appendices offer comparisons of the C89 and C99 standard “Kernighan & Ritchie” versions of the C language, together with a list of C’s standard library functions and other information. Students who use C after taking this course in the past will find this book useful as a source of help when writing C code. Thompson regarded it as a high-level, structured programming language, and it is excellent in this role. I am working on a large industrial project that uses C programming for the parts of the system that are intimately involved with its low-level interfaces. So C is alive and well despite successors such as C++, C# and Java, which impose and encourage a stricter discipline on programming style, often with an object-oriented flavour. Indeed, early versions of C compilers, designed in the early 1980s by Bjarne Stroustrup, compiled to C as the “low-level” target language, which could then itself be compiled to object code.

The first edition of this text, from 1996, covered the C89 standard of 1989; this second edition has been updated to follow the C99 standard of 1999, with differences clearly indicated. The text has also been thoroughly updated to take references to older operating systems such as DOS.

The volume is comprehensively, with more than 800 pages of material and a good index. Chapters include exercises, with online resources for instructors, facilitating its use as a textbook, as well as a work of reference. Appendices offer comparisons of the C89 and C99 standard “Kernighan & Ritchie” versions of the C language, together with a list of C’s standard library functions and other information. Students who use C after taking this course in the past will find this book useful as a source of help when writing C code. Thompson regarded it as a high-level, structured programming language, and it is excellent in this role. I am working on a large industrial project that uses C programming for the parts of the system that are intimately involved with its low-level interfaces. So C is alive and well despite successors such as C++, C# and Java, which impose and encourage a stricter discipline on programming style, often with an object-oriented flavour. Indeed, early versions of C compilers, designed in the early 1980s by Bjarne Stroustrup, compiled to C as the “low-level” target language, which could then itself be compiled to object code.

The first edition of this text, from 1996, covered the C89 standard of 1989; this second edition has been updated to follow the C99 standard of 1999, with differences clearly indicated. The text has also been thoroughly updated to take references to older operating systems such as DOS.

The volume is comprehensively, with more than 800 pages of material and a good index. Chapters include exercises, with online resources for instructors, facilitating its use as a textbook, as well as a work of reference. Appendices offer comparisons of the C89 and C99 standard “Kernighan & Ritchie” versions of the C language, together with a list of C’s standard library functions and other information. Students who use C after taking this course in the past will find this book useful as a source of help when writing C code.