Recommended Books on C/C++ Programming

Basic Books:

B. Kernighan & Ritchie. *The C Programming Language* (2nd edition), Prentice Hall, 1988.
This is the bible of the C language by the people who invented the language. It is a beautiful, condensed description of the language, with excellent examples, but not always easy to read. Any serious C/C++ programmer should have this book in his personal library. There are some glitches where the text is not fully up to the C89 Standard, so be sure to refer to Harbison & Steele (see below).

B. Stroustrup. *The C++ Progamming Language*, Addison Wesley (3rd edition) 1997. This is the bible on C++ by the man who designed the language. It is somewhere between a teaching text and a reference, and it is often not easy reading until you have learned the basics of the C++ language. However, this book should be in the library of any serious C++ programmer, and it should be read often! Be sure to get the third edition - earlier editions are obsolete.

S.B. Lippman & J. Lajoie. *C++ Primer* (3rd edition), Addison-Wesley 1998
Pretty meaty for a "primer." My favorite textbook-style introduction to C++. Very large, covers most of the language with good explanations and examples. An excellent place to look things up if Stroustrup seems unclear.

Reference Books:

S. P. Harbison & G.L. Steele, *C: A Reference Manual* (5th edition), Prentice Hall, 2002.
This is an excellent reference for the C language which covers all of the standard C libraries with descriptions of each of the library functions. It is truly a reference - not a book which you read from cover to cover as a text. Note that many of the standard C library functions are also called by C++ programs, so this reference is valuable for both C and C++ programmers. Earlier editions will still be useful, but the most recent covers C99.

Josuttis, N. *The C++ Standard Library: A Tutorial and Reference*. Addison-Wesley, 1999. An excellent source on the whole Standard Library, with very good examples, explanations, and suggestions. Oriented toward how-to-use, rather than how-it-works, for which the old Nelson book (below) can be useful.

Books of Advice:
Sutter, H., and Alexandrescu, A. *C++ Coding Standards: 101 Rules, Guidelines, and Best Practices*. Addison-Wesley, 2004. If you get only one book of advice, get this one. It pulls a lot of stuff together from a variety of sources (and provides the pointers).

S.C. Dewhurst, *C++ Gotchas: Avoiding Common Problems in Coding and Design*. Addison-Wesley, 2002.
A very useful book. Covers a wide range of problems, so good even for the beginner.

S. Meyers, *Effective C++* (3rd edition). Addison-Wesly, 2005
S. Meyers, *More Effective C++*. Addison-Wesley. 1996.
S. Meyers, *Effective STL*. Addison-Wesley. 2001.
Outstanding books, but most useful only after you been introduced to the basics. A series of brief articles about important concepts or techniques presented in the form of guidelines. Example "Use const wherever possible."

Books about OO Design:

A. Shalloway & J. Trott, *Design Patterns Explained: A New Perspective on Object-Oriented Design***.** Addison-Wesley, 2001.
A readable discussion of how to use design patterns. A bit chatty, but a good introduction.

E. Gamma, R. Helm, R. Johnson, J. Vlissides, *Design Patterns: Elements of reusable object-oriented software*. Addison-Wesley, 1995.
The book that introduced design patterns to the programming world. After OOP became widespread, these wizards noticed that certain patterns of classes were being used repeatedly - they document them along with pointers about when and how to use these patterns. An advanced book, but saves reinventing the wheel!

M. Fowler, K. Beck , J.Brant, & W. Opdyke. *Refactoring: Improving the Design of Existing Code*. Addison-Wesley, 1999.
Read this book to learn how improve your code in both big ways and small, even while you are writing it.

A.H. Riel, *Object-Oriented Design Heuristics*. Addison-Wesley, 1996.
A bit old, but uniquely useful. Explains practical matters of OO design in terms of a collection of guidelines, with good discussion of tradeoffs.

Advanced Books:
B. Stroustrup, *The Design & Evolution of C++*. Addison-Wesley, 1994.
Best book about why C++ is the way it is - makes sense of many language features, and reveals much about the world of professional programmers.

S. Lippman, *Inside the C++ Object Model*. Addison-Wesley, 1996.
Best book about how C++ works. A readable explanation about what goes on "under the hood" with topics like virtual function calls, function overloading, and so forth. Increases wisdom.

A. Alexandrescu, *Modern C++ Design*. Addison-Wesley, 2001. The title understates the content. It should be "Mind-bending C++ Design." State-of-the-art techniques based on using templates to do amazing things. Goes far beyond Stroustrup, so hang on. This is at the cutting edge of programming techniques.

H. Sutter, *Exceptional C++: 47 Engineering Puzzles, Programming Problems, and Exception-Safety Solutions*. Addison-Wesley, 1999.
H. Sutter, *More Exceptional C++: 40 Engineering Puzzles, Programming Problems, and Solutions*. Addison-Wesley, 2001.
Presents many important ideas and techniques in puzzle form, based on the "Guru of the Week" feature of the newsgroup comp.lang.c ++.moderated. You can get a lot out of it even if you hate solving puzzles and just skip to the answers.

Other Useful Books:

B. Kernighan & B. Pike. *The Practice of Programming*. Addison-Wesley, 1999.
Presents a broad range of valuable practical advice, using examples in C, C++, Java, and Perl. Tends to be Unix-oriented, but still an excellent source of advice.

Nelson, M. *C++ Programmer's Guide to t he Standard Template Library*. IDG Books, 1995.
An obsolete, but especially readable and thorough explanation of how the STL works (including the guts of it) and how to use it. This dates from the pre-Standard days. If you find it for sale cheap, you might pick up a copy, but always check against Stroustrup or a more up to date reference, such as Josuttis.