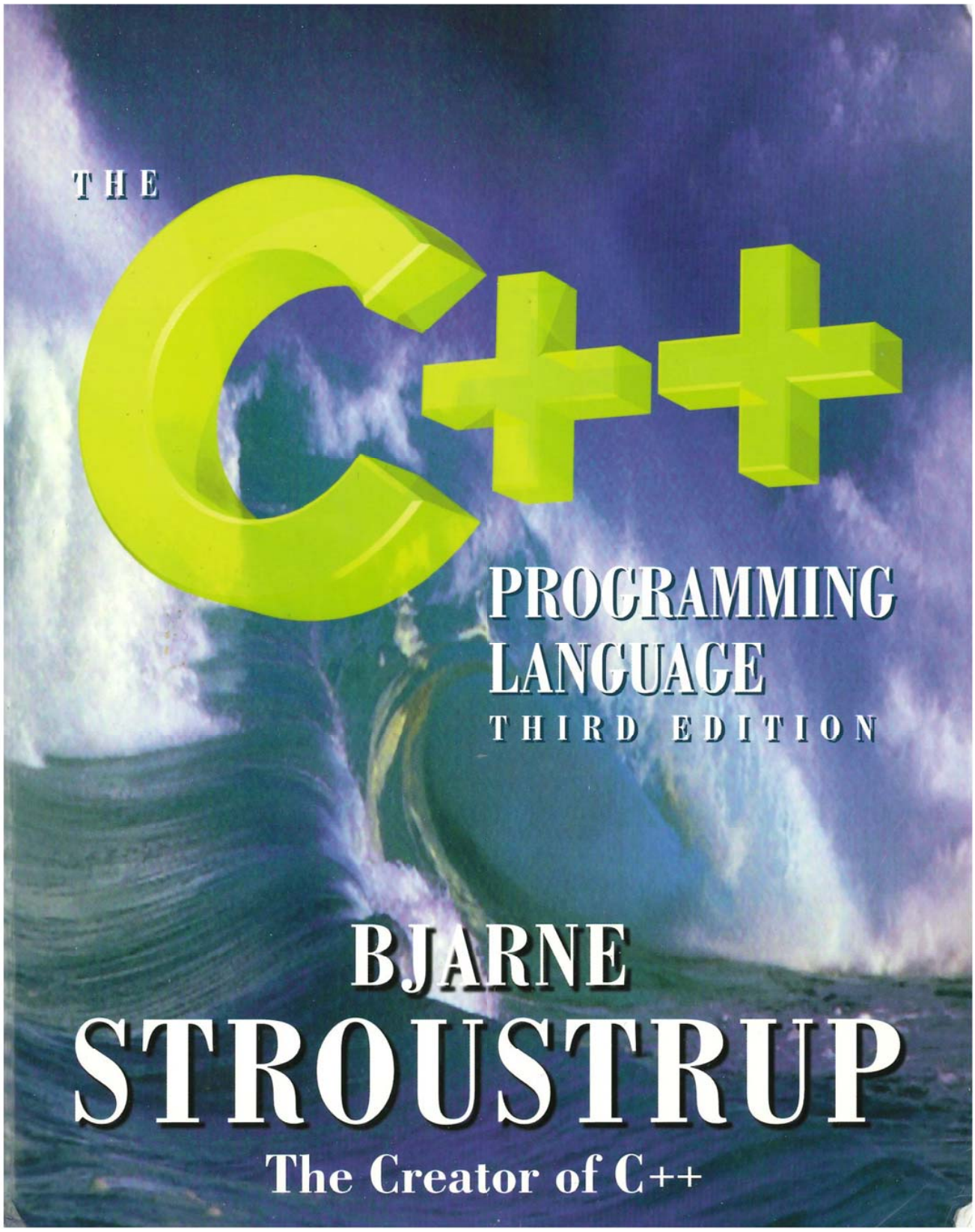

The C++ Programming Language

Third Edition

Bjarne Stroustrup

AT&T Labs
Murray Hill, New Jersey

Addison-Wesley
An Imprint of Addison Wesley Longman, Inc.

Reading, Massachusetts · Harlow, England · Menlo Park, California
Berkeley, California · Don Mills, Ontario · Sydney
Bonn · Amsterdam · Tokyo · Mexico City

The publisher offers discounts on this book when ordered in quantity for special sales. For more information please contact:

Corporate & Professional Publishing Group
Addison-Wesley Publishing Company
One Jacob Way
Reading, Massachusetts 01867

Library of Congress Cataloging-in-Publication Data

Stroustrup, Bjarne
The C++ Programming Language / Bjarne Stroustrup. — 3rd. ed.
p. cm.
Includes index.
ISBN 0-201-88954-4
1. C++ (Computer Programming Language) I. Title
QA76.73.C153S77 1997 97-20239
005.13'3—dc21 CIP

This book was typeset in Times and Courier by the author.

ISBN 0-20-88954-4
Printed on recycled paper
5 6 7 8 9 10 CRS 01 00 99 98
5th Printing January, 1998

4

Types and Declarations

Accept nothing short of perfection!
– anon

Perfection is achieved
only on the point of collapse.
– C. N. Parkinson

Types — fundamental types — Booleans — characters — character literals — integers — integer literals — floating-point types — floating-point literals — sizes — ***void*** — enumerations — declarations — names — scope — initialization — objects — ***typedef***s — advice — exercises.

4.1 Types

Consider

```
x = y+f(2);
```

For this to make sense in a C++ program, the names *x*, *y*, and *f* must be suitably declared. That is, the programmer must specify that entities named *x*, *y*, and *f* exist and that they are of types for which = (assignment), + (addition), and ( ) (function call), respectively, are meaningful.

Every name (identifier) in a C++ program has a type associated with it. This type determines what operations can be applied to the name (that is, to the entity referred to by the name) and how such operations are interpreted. For example, the declarations

```
float x;        // x is a floating-point variable
int y = 7;      // y is an integer variable with the initial value 7
float f(int);   // f is a function taking an argument of type int and returning a floating-point number
```

would make the example meaningful. Because *y* is declared to be an ***int***, it can be assigned to, used in arithmetic expressions, etc. On the other hand, *f* is declared to be a function that takes an ***int*** as its argument, so it can be called given a suitable argument.

This chapter presents fundamental types (§4.1.1) and declarations (§4.9). Its examples just demonstrate language features; they are not intended to do anything useful. More extensive and realistic examples are saved for later chapters after more of C++ has been described. This chapter simply provides the most basic elements from which C++ programs are constructed. You must know these elements, plus the terminology and simple syntax that goes with them, in order to complete a real project in C++ and especially to read code written by others. However, a thorough understanding of every detail mentioned in this chapter is not a requirement for understanding the following chapters. Consequently, you may prefer to skim through this chapter, observing the major concepts, and return later as the need for understanding of more details arises.

4.1.1 Fundamental Types

C++ has a set of fundamental types corresponding to the most common basic storage units of a computer and the most common ways of using them to hold data:

§4.2 A Boolean type (***bool***)
§4.3 Character types (such as ***char***)
§4.4 Integer types (such as ***int***)
§4.5 Floating-point types (such as ***double***)

In addition, a user can define

§4.8 Enumeration types for representing specific sets of values (***enum***)

There also is

§4.7 A type, ***void***, used to signify the absence of information

From these types, we can construct other types:

§5.1 Pointer types (such as ***int****)
§5.2 Array types (such as ***char***[])
§5.5 Reference types (such as ***double***&)
§5.7 Data structures and classes (Chapter 10)

The Boolean, character, and integer types are collectively called *integral types*. The integral and floating-point types are collectively called *arithmetic types*. Enumerations and classes (Chapter 10) are called *user-defined types* because they must be defined by users rather than being available for use without previous declaration, the way fundamental types are. In contrast, other types are called *built-in types*.

The integral and floating-point types are provided in a variety of sizes to give the programmer a choice of the amount of storage consumed, the precision, and the range available for computations (§4.6). The assumption is that a computer provides bytes for holding characters, words for holding and computing integer values, some entity most suitable for floating-point computation, and addresses for referring to those entities. The C++ fundamental types together with pointers and arrays present these machine-level notions to the programmer in a reasonably implementation-independent manner.

For most applications, one could simply use ***bool*** for logical values, ***char*** for characters, ***int*** for integer values, and ***double*** for floating-point values. The remaining fundamental types are

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.