

TURBO C++



001-6424
POH

IRA POHL

TURBO C++

IRA POHL

University of California
Santa Cruz

814
SEMINAR LIBRARY
Department of Computer Science
UNIVERSITY OF KARACHI
7-8-96

A GIFT OF
THE ASI FOUNDATION
BOOKS FOR ASIA
SAN FRANCISCO, CALIFORNIA, USA
NOT FOR SALE



The Benjamin/Cummings Publishing Company, Inc.

Redwood City, California ■ Fort Collins, Colorado
Menlo Park, California ■ Reading, Massachusetts ■ New York
Don Mills, Ontario ■ Wokingham, U.K. ■ Amsterdam ■ Bonn
Sydney ■ Singapore ■ Tokyo ■ Madrid ■ San Juan

CONTENTS

0 OBJECT-ORIENTED PROGRAMMING IN TURBO C++ 1

- 0.1 Object-Oriented Programming 2
- 0.2 Why C++ Is a Better C 3
- 0.3 Why Switch to Turbo C++? 5
- 0.4 References 5

1 AN OVERVIEW OF C++ AND OBJECT- ORIENTED PROGRAMMING 7

- 1.1 Output 8
- 1.2 Input 10
- 1.3 Function Prototypes 13
- 1.4 Classes and Abstract Data Types 14
- 1.5 Overloading 16
- 1.6 Constructors and Destructors 19
- 1.7 Object-Oriented Programming and
Inheritance 22
- 1.8 Turbo C++ Considerations 24
- Summary 27
- Exercises 29

2 C++ AS A BETTER C 33

- 2.1 Comment Style 34
- 2.2 Avoiding the Preprocessor: inline
and const 35

2.3	Declarations	36	
2.4	The Uses of <code>void</code>	40	
2.5	Scope Resolution Operator <code>::</code>		42
2.6	Function Prototypes	43	
2.7	Call-by-Reference and Reference Declarations	45	
2.8	Default Arguments	46	
2.9	Overloading Functions	47	
2.10	Free Store Operators <code>new</code> and <code>delete</code>	48	
2.11	Odds and Ends	51	
2.12	Turbo C++ Considerations		52
	Summary	55	
	Exercises	57	

3

CLASSES

63

3.1	The Aggregate Type <code>struct</code>	64
3.2	Structure Pointer Operator	66
3.3	An Example: Stack	67
3.4	Member Functions	70
3.5	Visibility <code>private</code> and <code>public</code>	73
3.6	Classes	75
3.7	<code>static</code> Member	76
3.8	Nested Classes	77
3.9	An Example: Flushing	77
3.10	Turbo C++ Considerations	82
	Summary	84
	Exercises	85

4

CONSTRUCTORS AND DESTRUCTORS

91

4.1	Classes with Constructors	92
4.2	Classes with Destructors	95
4.3	An Example: Dynamically Allocated Strings	95
4.4	A Class <code>vect</code>	99
4.5	Members That Are Class Types	101
4.6	An Example: A Singly Linked List	102
4.7	Two-Dimensional Arrays	106
4.8	The <code>this</code> Pointer	108
4.9	Turbo C++ Considerations	108
	Summary	111
	Exercises	113

5

OPERATOR OVERLOADING AND CONVERSIONS 119

5.1	The Traditional Conversions	120
5.2	ADT Conversions	122
5.3	Overloading and Function Selection	124
5.4	Friend Functions	127
5.5	Overloading Operators	130
5.6	Unary Operator Overloading	131
5.7	Binary Operator Overloading	133
5.8	Overloading Assignment and Subscripting Operators	135
5.9	Turbo C++ Considerations	139
	Summary	141
	Exercises	143

6 INHERITANCE 149

6.1	A Derived Class	150
6.2	<code>public</code> , <code>private</code> , and <code>protected</code>	152
6.3	A Derived Safe Array Type <code>vect_bnd</code>	153
6.4	Typing Conversions and Visibility	157
6.5	Virtual Functions	160
6.6	An Example: A Class Hierarchy	162
6.7	A Binary Tree Class	167
6.8	Turbo C++ Considerations	170
	Summary	172
	Exercises	173

7 INPUT/OUTPUT 179

7.1	The Output Class <code>ostream</code>	180
7.2	Formatted Output and <code>iomanip.h</code>	181
7.3	User-Defined Types: Output	183
7.4	The Input Class <code>istream</code>	185
7.5	Files	186
7.6	The Functions and Macros in <code>ctype.h</code>	190
7.7	Using Stream States	190
7.8	Turbo C++ Considerations	193
	Summary	194
	Exercises	196

8

ADVANCED FEATURES 201

8.1	Multiple Inheritance	202	
8.2	Abstract Base Classes	206	
8.3	Default Assignment and Overloading <i>new</i>	207	
8.4	Pointer Operators	208	
8.5	Scope and Linkage	211	
8.6	Iterators	213	
8.7	An Example: Word Frequency	215	
8.8	OOP: Object-Oriented Programming	221	
8.9	Platonism: Object-Oriented Design	222	
8.10	Turbo C++ Considerations	223	
	Summary	224	
	Exercises	226	

APPENDIX A	ASCII Character Codes	229
APPENDIX B	Operator Precedence and Associativity	231
APPENDIX C	Turbo C++ Language Guide	232
APPENDIX D	Turbo C++ Editor Commands	267
INDEX		271

Object-Oriented Programming in Turbo C++

-
- 0.1 Object-Oriented Programming
 - 0.2 Why C++ Is a Better C
 - 0.3 Why Switch to Turbo C++?
 - 0.4 References
-

C++ was created by Bjarne Stroustrup in the early 1980s. Stroustrup had two main goals: (1) C++ was to be compatible with ordinary C, and (2) it was to extend C using the class construct of Simula 67. The class construct is an extension of the C `struct`. The language, in an early form, is described in B. Stroustrup, *The C++ Programming Language* [1]. Important additions to the current language are described in B. Stroustrup, *The Evolution of C++: 1985 to 1987* [2].