



C++

Programming Step-by-Step

Asadullah Shah



IIUM PRESS

INTERNATIONAL ISLAMIC UNIVERSITY MALAYSIA

C++ PROGRAMMING: STEP BY STEP

Editors

Asadullah Shah



IIUM Press

Published by:
IIUM Press
International Islamic University Malaysia

First Edition, 2011
©IIUM Press, IIUM

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without any prior written permission of the publisher.

Perpustakaan Negara Malaysia

Cataloguing-in-Publication Data

Bibliography p.
Includes Index
ISBN

ISBN: 978-967-418-090-4

Member of Majlis Penerbitan Ilmiah Malaysia - MAPIM
(Malaysian Scholarly Publishing Council)

Printed by :
IIUM PRINTING SDN. BHD.
No. 1, Jalan Industri Batu Caves 1/3
Taman Perindustrian Batu Caves
Batu Caves Centre Point
68100 Batu Caves
Selangor Darul Ehsan

CONTENTS

DEDICATION	iii
PREFACE	viii
ACKNOWLEDGEMENT	ix
1. INTRODUCTION	
<i>Asadullah Shah and Assadullah Shaikh</i>	1
2. ARITHMETIC EXPRESSIONS AND DATA TYPES IN C++	
<i>Asadullah Shah and Assadullah Shaikh</i>	5
3. SENDING THE OUTPUT TO A PRINT FILE	
<i>Asadullah Shah and Assadullah Shaikh</i>	11
4. DECISION MAKING: IF-ELSE STATEMENTS AND RELATIONAL OPERATORS	
<i>Asadullah Shah and Assadullah Shaikh</i>	17
5. LOGICAL OPERATORS AND SWITCH STATEMENTS	
<i>Asadullah Shah and Assadullah Shaikh</i>	25
6. REVIEW, SUMMARY & BUILDING SKILL	
<i>Asadullah Shah and Khamran Khowaza</i>	33
7. ITERATIVE STRUCTURES	
<i>Asadullah Shah and Khamran Khowaza</i>	39

8. THE FOR LOOP

Asadullah Shah and Khamran Khowaza 49

9. THE DO-WHILE LOOP

Asadullah Shah and Khamran Khowaza 55

10. REVIEW OF VARIABLES, FORMATTING

Asadullah Shah and Khamran Khowaza 59

11. REVIEW OF ITERATIVE STRUCTURES

Asadullah Shah and Sumbul Khowaza 63

12. POST-TEST AND NESTED LOOPS

Asadullah Shah and Sumbul Khowaza 73

13. FUNCTIONS

Asadullah Shah and Sumbul Khowaza 83

14. CALL-BY-VALUE AND REFERENCE

Asadullah Shah and Sumbul Khowaza 91

15. MORE ON FUNCTIONS

Asadullah Shah and Sumbul Khowaza 99

16. STRUCTURES (STRUCT) AND FILES

Asadullah Shah and Muniba Shaikh 111

17. ARRAYS

Asadullah Shah and Muniba Shaikh 119

18. EXERCISE OF ARRAY

Asadullah Shah and Muniba Shaikh 127

19. READ DATA FROM A FILE	
<i>Asadullah Shah and Muniba Shaikh</i>	137
20. OBJECT ORIENTED PROGRAMMING	
<i>Asadullah Shah and Muniba Shaikh</i>	143
21. SELECTION SORTING	
<i>Asadullah Shah and Syed Ifthar Ali</i>	153
22. BUBBLE SORT ALGORITHM	
<i>Asadullah Shah and Syed Ifthar Ali</i>	161
23. REVIEW OF ARRAYS	
<i>Asadullah Shah and Syed Ifthar Ali</i>	167
24. LINEAR SEARCHING	
<i>Asadullah Shah and Syed Ifthar Ali</i>	179
25. BINARY SEARCH	
<i>Asadullah Shah and Syed Ifthar Ali</i>	189
26. VECTOR CLASS	
<i>Asadullah Shah and Ejaz Ahmed</i>	199
27. POINTERS	
<i>Asadullah Shah and Ejaz Ahmed</i>	203
28. FUNCTION POINTERS	
<i>Asadullah Shah and Ejaz Ahmed</i>	213
29. POLYMORPHISM AND VIRTUAL FUNCTIONS	
<i>Asadullah Shah and Ejaz Ahmed</i>	219

30. C++ REFERENCES	
<i>Asadullah Shah and Ejaz Ahmed</i>	223
31. CONST CORRECTNESS	
<i>Asadullah Shah and Osama Mahfooz</i>	229
32. MORE ON CONST KEYWORDS	
<i>Asadullah Shah and Osama Mahfooz</i>	235
33. GOTO STATEMENT	
<i>Asadullah Shah and Osama Mahfooz</i>	241
34. HANDLING ERRORS IN C++	
<i>Asadullah Shah and Osama Mahfooz</i>	249
35. STATIC: THE MULTIPURPOSE KEYWORD	
<i>Asadullah Shah and Osama Mahfooz</i>	253

31. CONST CORRECTNESS

Asadullah Shah and Osama Mahfooz

Department of Computer Science, Faculty of Information and
Communication Technology, International Islamic University Malaysia,
Malaysia

Abstract

The `const` keyword allows you to specify whether or not a variable is modifiable. You can use `const` to prevent modifications to variables and `const` pointers and `const` references prevent changing the data pointed to (or referenced).

31.1 Const Keyword

`Const` gives you the ability to document your program more clearly and actually enforce that documentation. By enforcing your documentation, the `const` keyword provides guarantees to your users that allow you to make performance optimizations without the threat of damaging their data. For instance, `const` references allow you to specify that the data referred to won't be changed; this means that you can use `const` references as a simple and immediate way of improving performance for any function that currently takes objects by value without having to worry that your function might modify the data. Even if it does, the compiler will prevent the code from compiling and alert you to the problem. On the other hand, if you didn't use `const` references, you'd have no easy way to ensure that your data wasn't modified.