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

Asadullah Shah and Assadullah Shaikh *1*

2. ARITHMETIC EXPRESSIONS AND DATA TYPES IN C++

Asadullah Shah and Assadullah Shaikh *5*

3. SENDING THE OUTPUT TO A PRINT FILE

Asadullah Shah and Assadullah Shaikh *11*

4. DECISION MAKING: IF-ELSE STATEMENTS AND RELATIONAL OPERATORS

Asadullah Shah and Assadullah Shaikh *17*

5. LOGICAL OPERATORS AND SWITCH STATEMENTS

Asadullah Shah and Assadullah Shaikh *25*

6. REVIEW, SUMMARY & BUILDING SKILL

Asadullah Shah and Khamran Khowaza *33*

7. ITERATIVE STRUCTURES

Asadullah Shah and Khamran Khowaza *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
Asadullah Shah and Muniba Shaikh *137*

20. OBJECT ORIENTED PROGRAMMING
Asadullah Shah and Muniba Shaikh *143*

21. SELECTION SORTING
Asadullah Shah and Syed Ifthar Ali *153*

22. BUBBLE SORT ALGORITHM
Asadullah Shah and Syed Ifthar Ali *161*

23. REVIEW OF ARRAYS
Asadullah Shah and Syed Ifthar Ali *167*

24. LINEAR SEARCHING
Asadullah Shah and Syed Ifthar Ali *179*

25. BINARY SEARCH
Asadullah Shah and Syed Ifthar Ali *189*

26. VECTOR CLASS
Asadullah Shah and Ejaz Ahmed *199*

27. POINTERS
Asadullah Shah and Ejaz Ahmed *203*

28. FUNCTION POINTERS
Asadullah Shah and Ejaz Ahmed *213*

29. POLYMORPHISM AND VIRTUAL FUNCTIONS
Asadullah Shah and Ejaz Ahmed *219*

30. C++ REFERENCES
Asadullah Shah and Ejaz Ahmed 223

31. CONST CORRECTNESS
Asadullah Shah and Osama Mahfooz 229

32. MORE ON CONST KEYWORDS
Asadullah Shah and Osama Mahfooz 235

33. GOTO STATEMENT
Asadullah Shah and Osama Mahfooz 241

34. HANDLING ERRORS IN C++
Asadullah Shah and Osama Mahfooz 249

35. STATIC: THE MULTIPURPOSE KEYWORD
Asadullah Shah and Osama Mahfooz 253

26. VECTOR CLASS

Asadullah Shah and Ejaz Ahmed
Department of Computer Science, Faculty of Information and Communication Technology, International Islamic University Malaysia, Malaysia

Abstract

Vectors are a kind of sequence container. As such, their elements are ordered following a strict linear sequence. Vector containers are implemented as dynamic arrays; Just as regular arrays, vector containers have their elements stored in contiguous storage locations, which means that their elements can be accessed not only using iterators but also using offsets on regular pointers to elements. But unlike regular arrays, storage in vectors is handled automatically, allowing it to be expanded and contracted as needed.

26.1 Vector Class

Here we have an array with names of a deck of card. We do not actually shuffle this array, instead we create an array of integers (swap[]) and place random numbers from 1 to 26 in it. Once we have the random numbers, we just display the card names in that order.

```
string cards[26]={
"CA","C2","C3","C4","C5","C6","C7","C8","C9","C10","CJ","CQ",
"CK",

"DA","D2","D3","D4","D5","D6","D7","D8","D9","D10","DJ","DQ
","DK"};

cout<<"\n\nHere is the swap deck: \n";
for (i=0;i <26;i++)
```