MCA : MANAGEMENT : SEMESTER-I

NEW SYLLABUS

C Programming with Data Structure

Dr. Mrs. Smita Chavan
Gautam Bapat
Shilpa Pawale

NIRALI PRAKASHAN
ADVANCEMENT OF KNOWLEDGE

A Text Book of

C PROGRAMMING WITH DATA STRUCTURE

FOR
M.C.A. : MANAGEMENT : SEMESTER - I
SUBJECT CODE : IT 12

AS PER NEW REVISED SYLLABUS FOR
MCA (PART I) FROM ACADEMIC YEAR 2015-2016

Dr. Mrs. SMITA CHAVAN
M.Sc. (Electronics), MCM, MCA, Ph.d. (Comp. Mgt.)
Associate Director
NBN – Sinhgad School of Computer Studies
Pune

GAUTAM BAPAT
M.C.A.
Asst. Professor, Computer Science & Applications
MITSOM College,
Pune

SHILPA PAWALE
M.C.S., M.C.A.
Lecturer, P.V.G.'s College of Science
Pune

Price ₹ 250.00

NIRALI PRAKASHAN
ADVANCEMENT OF KNOWLEDGE

N1192

C PROGRAMMING WITH DATA STRUCTURE (M.C.A. I) **ISBN 978-93-82448-06-8**

First Edition : September 2015

© : Authors

The text of this publication, or any part thereof, should not be reproduced or transmitted in any form or stored in any computer storage system or device for distribution including photocopy, recording, taping or information retrieval system or reproduced on any disc, tape, perforated media or other information storage device etc., without the written permission of Authors with whom the rights are reserved. Breach of this condition is liable for legal action.

Every effort has been made to avoid errors or omissions in this publication. In spite of this, errors may have crept in. Any mistake, error or discrepancy so noted and shall be brought to our notice shall be taken care of in the next edition. It is notified that neither the publisher nor the authors or seller shall be responsible for any damage or loss of action to any one, of any kind, in any manner, therefrom.

Published By :

NIRALI PRAKASHAN
Abhyudaya Pragati, 1312, Shivaji Nagar
Off J.M. Road, PUNE – 411005
Tel - (020) 25512336/37/39, Fax - (020) 25511379
Email : niralipune@pragationline.com

Printed By :

RACHANA OFFSETS
S. No. 15, Arihant Marg
Sukhsagar Nagar, Katraj
Pune - 411 046
Tel - (020) 26963220

☞ DISTRIBUTION CENTRES

PUNE

Nirali Prakashan : 119, Budhwar Peth, Jogeshwari Mandir Lane, Pune 411002, Maharashtra
Tel : (020) 2445 2044, 66022708, Fax : (020) 2445 1538
Email : bookorder@pragationline.com, niralilocal@pragationline.com

Nirali Prakashan : S. No. 28/27, Dhyari, Near Pari Company, Pune 411041
Tel : (020) 24690204 Fax : (020) 24690316
Email : dhyari@pragationline.com, bookorder@pragationline.com

MUMBAI

Nirali Prakashan : 385, S.V.P. Road, Rasdhara Co-op. Hsg. Society Ltd.,
Girgaum, Mumbai 400004, Maharashtra
Tel : (022) 2385 6339 / 2386 9976, Fax : (022) 2386 9976
Email : niralimumbai@pragationline.com

☞ DISTRIBUTION BRANCHES

JALGAON

Nirali Prakashan : 34, V. V. Golani Market, Navi Peth, Jalgaon 425001,
Maharashtra, Tel : (0257) 222 0395, Mob : 94234 91860

KOLHAPUR

Nirali Prakashan : New Mahadvar Road, Kedar Plaza, 1st Floor Opp. IDBI Bank
Kolhapur 416 012, Maharashtra. Mob : 9850046155

NAGPUR

Pratibha Book Distributors : Above Maratha Mandir, Shop No. 3, First Floor,
Rani Jhanshi Square, Sitabuldi, Nagpur 440012, Maharashtra
Tel : (0712) 254 7129

DELHI

Nirali Prakashan : 4593/21, Basement, Aggarwal Lane 15, Ansari Road, Daryaganj
Near Times of India Building, New Delhi 110002
Mob : 08505972553

BENGALURU

Pragati Book House : House No. 1, Sanjeevappa Lane, Avenue Road Cross,
Opp. Rice Church, Bengaluru – 560002.
Tel : (080) 64513344, 64513355,Mob : 9880582331, 9845021552
Email:bharatsavla@yahoo.com

CHENNAI

Pragati Books : 9/1, Montieth Road, Behind Taas Mahal, Egmore,
Chennai 600008 Tamil Nadu, Tel : (044) 6518 3535,
Mob : 94440 01782 / 98450 21552 / 98805 82331,
Email : bharatsavla@yahoo.com

niralipune@pragationline.com | www.pragationline.com

Also find us on www.facebook.com/niralibooks

PREFACE

There has been significant development in recent years in the field of Computer Science. The book is a perfect blend of technology which has been a field of dramatic revolution; this subject focuses on different technologies of it.

It gives us great pleasure in presenting this book **"C Programming with Data Structure"** designed to serve as a textbook for students of the First Semester of Master of Computer Application (M.C.A.), the book is organized in such a way that it mirrors the revised syllabus. The book will be found useful by a wide section of readers, teachers and students of Business, Technology and Computer Management courses in Indian Universities.

The book has its own unique features. It brings out the subject in a very simple and lucid manner for easy and comprehensive understanding of the basic concepts, its intricacies, procedures and practices. This book will help the readers to have a broader view on C Programming with Data Structure. The language used in this book is easy and will help students to improve their vocabulary of Technical terms and understand the matter in a better and happier way.

Particular attention has been paid to making this book stimulating and highly readable. The result is a text which is clear, focused and designed to capture student interest. This text is equally suitable for courses directed at undergraduates and postgraduates.

We have given our best inputs for this book. Any suggestions towards the improvement of this book and sincere comments are most welcome on niralipune@pragationline.com.

Authors

SYLLABUS

1. **An Overview of C**
   - 1.1 A Brief History of C
   - 1.2 Features and characteristics of C
   - 1.3 Structure of a 'C' Program
   - 1.4 Program Development Life Cycle
   - 1.5 Complier Vs Interpreters
   - 1.6 Compilation and Execution of C Program on DOS and UNIX, Linux
2. **Variables, Data Types, Operator and Expression**
   - 2.1 Character Set, C Tokens - Keywords and Identifiers Constants, Integer, Floating Point, Character, String, Enumeration
   - 2.2 Backslash characters / Escape sequences
   - 2.3 Data Types in C, Variables - Declaration and Definition, User - Defined Type declarations
   - 2.4 Operators and Expressions - Arithmetic, Relational, Logical, Increment, Decrement, Bit wise, Assignment, Conditional, Type conversions in Expressions - Implicit Type Conversion, Explicit Type Conversions
   - 2.5 Precedence and Associability of Operators
   - 2.6 Built in I/O Functions - Introduction, Console Input and Output functions, Formatted Input and Output (scanf/printf), sprintf and sscanf
3. **Control Statements**
   - 3.1 Introduction
   - 3.2 Selection Statements
   - 3.3 If, Nested if, if....else, else if Ladder
   - 3.4 ternary operator, switch, Nested switch, Conditional expression
   - 3.5 Iterative Statements - while loop, do-while loop, for loop, break and continue
   - 3.6 Jump Statements - Goto and label
   - 3.7 exit() function
   - 3.8 Compound Statements, Null Statements
4. **Array and String**
   - 4.1 Single Dimension Arrays - Declaration, Initialization, Accessing array Elements, Memory Representation
   - 4.2 Multidimensional Arrays - Declaration, Initialization, Accessing array Elements, Memory Representation.
   - 4.3 String (character array) - Declaration, Initialization, String Manipulation Functions.
5. **Pointers**
   - 5.1 Introduction- Basics of Pointer, Memory Organization, Application of Pointer, Declaration Of pointer, Initializing Pointer
   - 5.2 Pointer Expressions, De-referencing Pointer Void Pointer, Pointer Arithmetic
   - 5.3 Precedence of and, * operators , Pointer to Pointer, Constant Pointer,
   - 5.4 Pointers and Arrays, Pointers and character string, Array of pointers
   - 5.5 Dynamic Memory Allocation - sizeof(), malloc(), calloc(), realloc(), free()

6. Function

6.1 Introduction - Types of functions, Declaration and Definition, Arguments and local variables

6.2 Parameter passing – Call by value and Call by reference

6.3 Passing arrays, strings to functions, Pointers to functions

6.4 Recursion

7. Structure, Union, Enumeration and typedef

7.1 Structures - Declaration and Initializing Structure, Accessing Structure members, Structure Assignments, Array of Structures, Nested structure, Passing Structure to function, Structure Pointer, typedef keyword

7.2 Unions - Declaration and Initializing Union

7.3 Accessing union members, Difference between Structure and Union, Enumerated data type

8. Introduction to File Handling

8.1 Introduction

8.2 Opening a File Closing a File

8.3 Input/Output Operations on Files

8.4 Error Handling During I/O Operation

8.5 Random Access To Files

9. Searching and Sorting

9.1 Linear search and Binary search

9.2 Sorting- Selection sort, Insertion sort, Bubble sort

10. Basics of Data Structure

10.1 Data Structure

10.2 Implementation of Data Structure

11. Array as Data Structure

11.1 Storage Representation of Arrays

11.2 Applications of Arrays

11.3 Polynomial Representation Using Arrays
Addition of Two Polynomial
Multiplication of Two Polynomial

11.4 Sparse Matrices
Addition of Sparse Matrices
Transpose of a Sparse Matrix

12. Stack

12.1 Introduction

12.2 Definition

12.3 Operation on Stack

12.4 Static Implementation of a Stack

12.5 Application of Stack

12.6 Recursion

12.7 Infix, Prefix & Postfix expression

13. Queue

13.1 Introduction
13.2 Definition of a Queue
13.3 Operation on a Queue
13.4 Static Implementation of Queue
13.5 Types of Queue - Circular Queue, Priority Queue
13.6 DEQueue
13.7 Application of Queue
13.8 Reversing Stack using Queue

CONTENTS

Chapter 1...

AN OVERVIEW OF C

Contents ...

1.1 Introduction to 'C' Language

- C language is a general purpose and structured programming language developed by 'Dennis Ritchie' at AT &T's Bell Laboratories in the 1972s in USA.
- A structured programming language offers a variety of programming possibilities and capabilities. It support different control loops, different statements etc.
- C language combines the best elements of high-level language with control and flexibility of assembly language so it is a middle level language.
- C language is also called as 'Procedure oriented programming language.'
- C has now become a widely used professional language for following reasons :
  1. C language is easy to learn.
  2. C language is structured language.
  3. C language produces efficient programs.
  4. C language can handle low-level activities.
  5. C language can be compiled on a variety of computers.

- As a programming language, C is rather like **Pascal** or **Fortran** i.e. values are stored in variables programs are structured by defining and calling functions, program flow is controlled using loops, if statements and function calls. Input and output can be directed to the terminal or to files and related data can be stored together in arrays or structures. C allows the most precise control of input and output.

1.2 A Brief History of C

- The development of C was a cause of evolution of programming languages like Algol 60, CPL (Combined Programming Language), BCPL (Basic Combined Programming Language) and B.
- Table 1.1 shows brief history of C language.
- C was developed by Dennis Ritchie at Bell Laboratories in 1972. Most of its principles and ideas were taken from the earlier language B, BCPL and CPL.
- CPL was developed jointly between the Mathematical Laboratory at the University of Cambridge and the University of London Computer Unit in 1960s.
- CPL (Combined Programming Language) was developed with the purpose of creating a language that was capable of both machine independent programming and would allow the programmer to control the behavior of individual bits of information. But the CPL was too large for use in many applications.
- In 1967, BCPL (Basic Combined Programming Language) was created as a scaled down version of CPL while still retaining its basic features. This process was continued by Ken Thompson. He made B Language during working at Bell Labs.
- B Language was a scaled down version of BCPL. B Language was written for the systems programming.
- In 1972, a co-worker of Ken Thompson, Dennis Ritchie developed C Language by taking some of the generality found in BCPL to the B language.
- The original PDP-11 version of the Unix system was developed in assembly language.
- In 1973, C language had become powerful enough that most of the Unix kernel was rewritten in C. This was one of the first operating system kernels implemented in a language other than assembly.
- The 1989 standard for C is commonly referred as C89.
- During 1990s the development of C++ language consumed most programmers attention but work on C continued quietly along with a new standard for C being developed and the result was the 1999 standard for C, usually referred as C99. It retained all features of C89.

Table 1.1

Year of establishment	Language name	Description
1960	ALGOL-60	ALGOL is an acronym for Algorithmic Language developed by Cambridge University. It was the first structured procedural programming language, developed in the late 1950s and once widely used in Europe. But it was too abstract and too general structured language.
1963	CPL (Combined Programming Language)	CPL is an acronym for Combined Programming Language. It was developed at Cambridge University.
1967	BCPL (Basic Combined Programming Language)	BCPL is an acronym for Basic Combined Programming Language. It was developed by Martin Richards at Cambridge University in 1967. BCPL was not so powerful. So, it was failed.
1970	B	B language was developed by Ken Thompson at AT & T Bell Laboratories in 1970. It was machine dependent. So, it leads to specific problems.
1972	C	'C' Programming Language was developed by Dennis Ritchie at AT & T Bell Laboratories in 1972. This is general purpose, compiled, structured programming language. Dennis Ritchie studied the BCPL, then improved and named it as 'C' which is the second letter of BCPL.
1989	C89	The ANSI C developed C89 language and add some new library functions.
1999	C99	It retained nearly all of the features of C89, thus C is still C.

1.3 Features and Characteristics of C

1.3.1 Features of C

- C language consist of following features :
  1. **Middle Level Language :** C is a middle level language as it combines elements of high-level language with the functional of assembly language. C allows direct manipulation of bits, bytes, words, and pointers.

2. **Block Structured Language :** C is referred as a structured language because it is similar in many ways to other structured languages like ALGOL, Pascal and the likes. C allows compartmentalization of code and data. This is a distinguishing feature of any structured language.
3. **Code Portability :** The code written in C is machine independent which means, there is no change in 'C' instructions, when you change the Operating System or Hardware. There is hardly any change required to compile when you move the program from one environment to another.
4. **Recursion :** A function may call itself again and again this feature is called as recursion, is supported by C.
5. **Efficiency :** C provides fast program execution.
6. **High level language feature :** This feature allows the programmer to concentrate on the logic flow of the code.
7. **Low level features :** C has a close relationship with assembly languages. So it is easy to make assembly program in C.
8. **Powerful :** C is very powerful language since low level commands have been access like assembly language.
9. **Flexibility :** In C language programmer has many ways to accomplish the same task.
10. **Small size :** C language provides no input output facilities. This helps to keep program small.

1.3.2 Characteristics of C

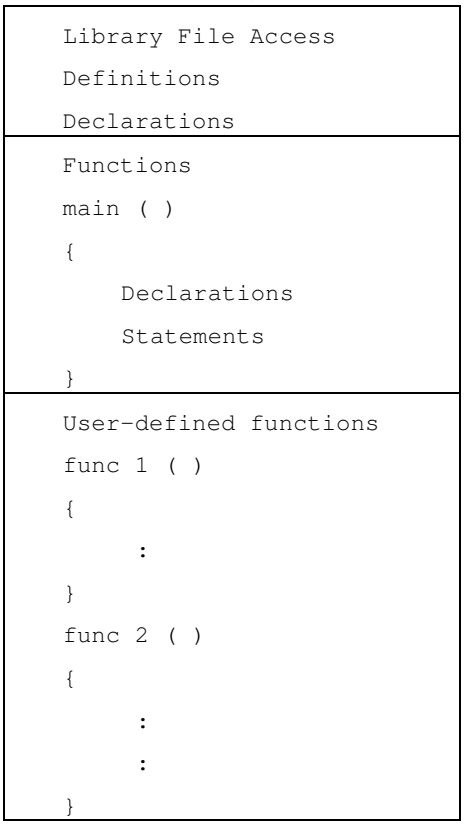
- The C language also exhibits the following specific characteristics :

1. There are a small, fixed number of keywords, including a full set of flow of control primitives : for, if, while, switch, and do..while. There is basically one namespace, and user-defined names are not distinguished from keywords by any kind of sigil.
2. There are a large number of arithmetical and logical operators, such as +, +=, ++, &, ~, etc.
3. More than one assignment may be performed in a single statement.
4. Function return values can be ignored when not needed.
5. Typing is static, but weakly enforced : all data has a type, but implicit conversions can be performed; for instance, characters can be used as integers.
6. Declaration syntax mimics usage context. C has no "define" keyword; instead, a statement beginning with the name of a type is taken as a declaration. There is no "function" keyword; instead, a function is indicated by the parentheses of an argument list.

7. Low-level access to computer memory is possible by converting machine addresses to typed pointers.
8. Procedures (subroutines not returning values) are a special case of function, with a dummy return type void.
9. Functions may not be defined within the lexical scope of other functions.
10. Function and data pointers permit ad hoc run-time polymorphism.
11. A preprocessor performs macro definition, source code file inclusion, and conditional compilation.
12. There is a basic form of modularity : files can be compiled separately and linked together, with control over which functions and data objects are visible to other files via static and extern attributes.
13. Complex functionality such as I/O, string manipulation, and mathematical functions are consistently delegated to library routines.

1.4 Structure of a C Program

- Following blocks shows structure of a C program.



- The C program begin executing at main( ). In the above declaration library file are used to give instructions to compiler for linking purpose.
- All constants and global variables declarations is done here. In second box consists of main( ). Local variable declarations and statements define in main( ).
- The last and third port consists of all user defined functions. These functions are called in main( ) function.

1.5 Program Development Life Cycle

- The program development life cycle of 'C' language has four steps as shown in Fig. 1.1.
  1. **Writing the source code :** The program is to be written using a text editor or word processor. Most language compilers have their built-in editors. The program or source code should be written in the syntax provided by that language. In C source code have extension .c.
  2. **Compiling the source code :** The source code is compiled using compiler of that language. The compiler converts the source code into the object code (in binary format). The compiler checks the syntax errors in the source code.
  3. **Linking the object code :** If the object code requires some precompiled library routines then the linking of the program with the library routines is done with the help of a linker. The linker creates a executable code file with extension **.exe**.
  4. **Executing the program :** Executable file is ready-to-run file. In DOS operating system we can run this file by just writing its name at prompt. In UNIX it is run using ./a.out (where **a.out** is executable file).

Fig. 1.1 : Program development life-cycle

1.5.1 A Simple C Program ("Hello World")

- Every C program must contain main( ) function. Execution of each and every C program starts with main function only.
- Every C program can contain more than one function but each program has to be a main( ) function in order to execute the program.
- C program is also called as **procedure oriented programming language** where importance is given to the procedure i.e. the task to be done which is expressed in the form of functions. So functions works as a building block for the programs.

- The program prints out "hello world" to the standard output, which is usually a terminal or screen display.

```
1. #include<stdio.h>
2. int main(void)
3. {
4. printf("hello world\n");
5. return 0;
6. }
```

1. `#include<stdio.h>`

This first line of the program is a preprocessing directive, **#include**. This causes the preprocessor — the first tool to examine source code as it is compiled — to substitute the line with the entire text of the **stdio.h** file. The header file **stdio.h** contains declarations for standard input and output functions such as **printf**.

2. `int main(void)`

This line indicates that a function named main is being defined. The main function serves a special purpose in C programs. The run-time environment calls the main function to begin program execution. The type specifier int indicates that the return value, the value of evaluating the main function that is returned to its invoker (in this case the run-time environment), is an integer. The keyword void as a parameter list indicates that the main function takes no arguments.

3. `{`

This opening curly brace indicates the beginning of the definition of the main function.

4. `printf("hello world\n");`

This line calls (executes the code for) a function named **printf**, which is declared in the included header stdio.h and supplied from a system library. In this call, the printf function is passed (provided with) a single argument, the address of the first character in the string literal "hello world\n". The semicolon (;) terminates the statement.

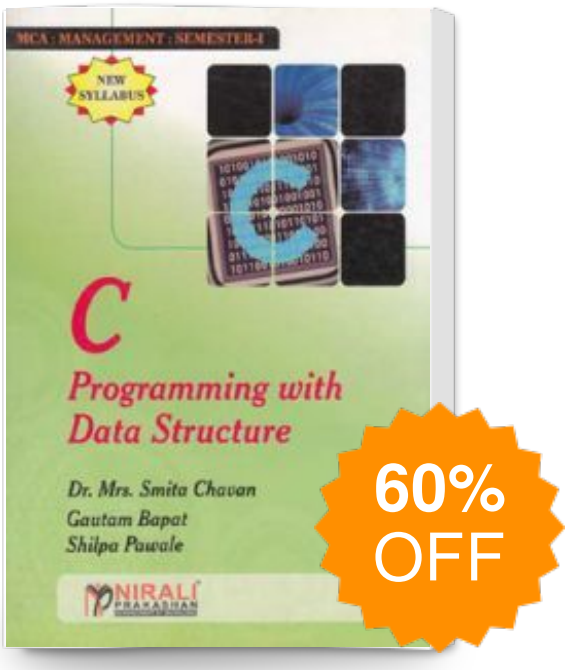
5. `return 0;`

This line terminates the execution of the main function and causes it to return the integer value 0, which is interpreted by the run-time system as an exit code, (indicating successful execution).

6. `}`

This closing curly brace indicates the end of the code for the main function.

C Programming With Data Structure



Publisher : Nirali Prakashan ISBN : 9789382448068 Author : Dr. Mrs. Smita Chavan, Gautam Bapat, Shilpa Pawale

Type the URL : http://www.kopykitab.com/product/19626

Get this eBook