



**PROGRAMMING WITH C**  
An Introduction

ALEX SIDAY

Edward Arnold

2.005.493-1

2-005-493-1

# Programming with **C**

## An Introduction

**Alex Siday**

University of Teesside



**Edward Arnold**

A member of the Hodder Headline Group  
LONDON NEW YORK SYDNEY AUCKLAND

# Contents

<b>Preface</b>	<b>v</b>
<b>1 The computer – a programmer's perspective</b>	<b>1</b>
Computers and programs. A simplified view of the hardware of a typical personal computer. Main memory and its uses. Binary numbers. The structures and organisations of main memory. The contents of memory. Memory addresses. Machine code programs and programming languages. The editor and text files. Syntax errors. Linking separately compiled C programs. Exercises	
<b>2 Program design</b>	<b>24</b>
Program specification. Program design. Example 1 – Ordering three integer values. Dry running algorithms. Analysing an algorithm's behaviour. Example 2 – Finding quotient and remainder of two numbers. Assignments and expressions. Operator precedence. Iterations and the while statement. Assignments, pre-conditions and post-conditions. Analysing iteration. Exercises	
<b>3 Turning algorithms into C programs</b>	<b>49</b>
Algorithms. The role of the semi-colon in C. The do statement. Declaring variables. C functions. Streams and standard input and output in C. Include files. Program testing. Converting digit sequences to numbers. Character constants. Generating a number from its digit characters. The conversion algorithm. C code for conversion. The conversion code in the form of a function. The general structure of a function declaration. Returning results from functions. Using <code>getint</code> . Processing text. Buffering keyboard input. The program algorithm. Control characters. Exercises	
<b>4 Modular program structure</b>	<b>76</b>
The program specification. Finding a suitable design. Detailed design and program implementation. An example – arithmetic expression evaluation. Dealing with addition and subtraction. <code>get_operator</code> . <code>get_operand</code> . Conditional expressions involving more than one operator. <code>evaluate_expression</code> . Conditional expressions. <code>evaluate_1</code> . Adding multiplication and division. The behaviour of the division operator <code>/</code> and decimals. Adding brackets. Exercises	

<b>5</b>	<b>Functions, parameters and pointers</b>	<b>105</b>
	Function arguments. Declaring functions that take arguments. Calling functions with arguments. The function remainder. Calling quotient and remainder from main. The allocation of memory to variables and parameters. Pointers. The function quot_rem. Function calls as statements and type void. External objects. Extent. Static extent. Local extent. Dynamic extent. Scope and linkage. Function prototypes. Defining declarations and referencing declarations. External references. Restricting accessibility using static. Exercises.	
<b>6</b>	<b>Structured data types</b>	<b>128</b>
	Arrays. Array elements. Array declarations. Array initialisations. Array indexes. Accessing arrays using the for statement. Increment and decrement operators. Arrays and pointers. Array arguments and array results. Strings. Reading in a new name. Creating a dynamic array. Copying strings. Structure types and structured values. Structure initialisations. Accessing the components of a structured variable using the selection operator. Treating structures as single entities. Using array types and structure types in arrays and structures. Files. Files and streams. Creating a file of integers. Accessing a file of integers. Exercises.	
<b>7</b>	<b>A telephone directory management system</b>	<b>163</b>
	Directory lookup. Directory update. Module implementations. Look up. The function read_line. The function find. Update directory. Linked lists. A linked list of integers. Structure component selection using ->. The function find. The function read_directory. delete entries. Exercises.	
<b>8</b>	<b>An assembly case study in C</b>	<b>193</b>
	The target machine. The format of instructions in memory. Assembler features. The assembler/interpreter system. The assembler. Enumerations. Dealing with labels. Lexical analysis. Label table management. Machine code generation. The modules error management and code management. The interpreter. The switch statement. Exercises.	
	<b>Appendix 1 The C language – an outline</b>	<b>218</b>
	Identifiers. Declarations. The basic types of values used within C programs. Dynamic objects. Constants. The declaratyion of scalar variables. Expressions. Lvalues and Rvalues. Precedence and associativity. The operators. Conditional expressions. Structured values. Array declarations. Struct declarations. Function declaration. Statements. The C Library.	
	<b>Appendix 2 The assembler/interpreter in Chapter 8</b>	<b>246</b>
	<b>Appendix 3 ASCII character codes</b>	<b>264</b>
	<b>Index</b>	<b>265</b>