Contents

Preface xvii
Acknowledgments xix
Organization of this Book xxi
Author Biography xxiii

1. Introduction to the Computer 1
1.1 Introduction 1
1.2 Definition and Characteristics of a Computer System 1
1.3 History of the Computer 2
1.4 Basic Computer Organization 4
1.4.1 Input Devices 5
1.4.2 Memory 6
1.4.3 Central Processing Unit 6
1.4.4 Output Devices 7
1.5 Computer Memory 7
1.5.1 Registers 8
1.5.2 Cache Memory 8
1.5.3 Primary Memory 8
1.5.4 Random Access Memory 10
1.5.5 Read Only Memory 10
1.5.6 Secondary Memory 10
1.5.7 Hard Disk Drive 10
1.5.8 Solid State Drive 11
1.6 Introduction to the Operating System 12
1.6.1 Hardware and Software 12
1.6.2 Operating System 12
1.6.3 Functions of an Operating System 13
1.7 Review Questions 14
1.7.1 Long Answers 14
1.7.2 Short Answers 14
1.7.3 Practical Exercises 15
References 16

2. Number Systems 17
2.1 Introduction 17
2.1.1 Non-positional Number System 17
2.1.2 Positional Number System 17
2.2 Positional Number Systems 18
2.2.1 Decimal Number System 18
2.2.2 Binary Number System 20
2.2.3 Hexadecimal Number System 21
2.2.4 Octal Number System 22

2.3 Number Conversion 23
2.3.1 Binary to Decimal 23
2.3.1.1 Approach 1 24
2.3.1.2 Approach 2 24
2.3.2 Binary Fraction to Decimal Conversion 25
2.3.3 Binary to Decimal Conversion 26
2.3.4 Decimal Fraction to Binary Fraction 27
2.3.5 Decimal to Octal Conversion 28
2.3.6 Octal to Decimal Conversion 29
2.3.7 Octal to Binary Conversion 29
2.3.7.1 Procedure 1 29
2.3.7.2 Procedure 2 30
2.3.8 Binary to Octal Conversion 30
2.3.8.1 Procedure 1 30
2.3.8.2 Procedure 2 31
2.3.9 Decimal to Hexadecimal Conversion 32
2.3.10 Hexadecimal to Decimal Conversion 33
2.3.11 Hexadecimal to Binary Conversion 33
2.3.11.1 Procedure 1 33
2.3.11.2 Procedure 2 34
2.3.12 Binary to Hexadecimal Conversion 34
2.3.12.1 Procedure 1 34
2.3.12.2 Procedure 2 35
2.4 Review Questions 36
2.4.1 Conversion Questions 36

3. Problem Solving through Flowcharts and Algorithms 39
3.1 Introduction 39
3.2 Problem-solving Approach 40
3.3 Algorithm Design 41
3.3.1 Characteristics of an Algorithm 42
3.4 Basics of an Algorithm 43
3.4.1 Advantages of Using an Algorithm 44
3.4.2 Example: Write an Algorithm to Add Two Numbers and Produce the Sum 44
3.4.3 Algorithm 3.1 45
3.5 Flowcharts 45
3.5.1 Advantages of Using a Flowchart 45
3.5.2 Flowchart Symbols 46
3.5.3 Flowchart Drawing Guidelines 46
3.6 Example Problems 48
3.7 Basics of a Programming Language 53
3.7.1 Low-level Languages 53
3.7.1.1 Machine-level Languages 54
3.7.1.2 Assembly-level Languages 54
3.7.2 High-level Languages 55
3.7.2.1 Compiler vs. Interpreter 57
3.7.2.2 Advantages 57

3.8 Review Questions 57
3.8.1 Objective Type Questions 57
3.8.2 Practice Problems 57
3.8.3 Subjective Questions 58
Reference 59

4. Introduction to C Programming 61
4.1 Introduction 61
4.2 History of C 62
4.3 Executing a C Program 64
4.3.1 Editing 64
4.3.2 Compiling 64
4.3.3 Linking 65
4.3.4 Executing 65
4.4 Structure of a C Program 65
4.4.1 Documentation 65
4.4.2 Header Files 65
4.4.3 Global Variables 66
4.4.4 `main()` Function 66
4.4.5 Subprograms 67
4.4.6 Your First C Program 67
4.5 Compilers and Editors for Executing C Programs 69
4.5.1 Editors 69
4.5.2 Compilers 69
4.5.3 Executing Your First C Program 71
4.5.3.1 Mac 71
4.5.3.2 Windows 72
4.5.3.3 Linux 72
4.6 Review Questions 73
4.6.1 Objective Questions 73
4.6.2 Short Answer Questions 73
4.6.3 Programming Questions 73
4.6.4 Long Questions 75
References 75

5. Constants, Variables, and Data Types 77
5.1 Introduction 77
5.2 C Character Sets 77
5.3 Keywords 78
5.4 Variables and Identifiers 79
5.5 Data Types 80
5.5.1 Primary Data Types 81
5.5.2 Integer Data Types 81
5.5.3 Floating Point Types 82
5.5.4 Character Data Types 83
5.5.5 Void Types 83
5.6 Declaration of Variables 84
5.7 Constants 86
5.7.1 Integer Constants 86

5.7.2 Real Constants 87
5.7.3 Fractional Form 87
5.7.4 Exponential Form 87
5.7.5 Character Constants 87
5.7.6 String Constants 87
5.8 Learn to Code Examples 88
5.9 Escape Sequences 91
5.10 Review Questions 92
5.10.1 Objective Questions 92
5.10.2 Programming Questions 92
5.10.3 Subjective Questions 94

6. Operators and Expressions 95
6.1 Introduction 95
6.2 Arithmetic Operators 96
6.3 Relational Operators 97
6.4 Assignment Operators 98
6.5 Logical Operators 99
6.6 Increment and Decrement Operators 100
6.7 Conditional Operators 103
6.7.1 Nested Conditional Operators 105
6.8 Bitwise Operators 105
6.8.1 Bitwise AND, OR, XOR 106
6.8.2 One's Complement (~) Operator 107
6.8.3 Two's Complement Representation 107
6.8.4 Left Shift Operator (≪) and Right Shift Operator (≫) 109
6.9 Special Operators 112
6.9.1 The Comma Operator 112
6.9.2 The sizeof Operator 113
6.10 Expressions 113
6.10.1 Evaluation of Expressions 114
6.10.2 Rules for Evaluation of Expressions 114
6.11 Type Conversion 115
6.11.1 Implicit Type Casting 115
6.11.2 Explicit Type Conversion 116
6.12 Operator Precedence and Associativity 116
6.13 Review Questions 118
6.13.1 Objective Type Questions 118
6.13.2 Programming Questions 119
6.13.3 Subjective Type Questions 120

7. Basic Input/Output 123
7.1 Introduction 123
7.2 Unformatted Functions 124
7.2.1 `getchar()` and `putchar()` 124
7.2.2 `gets()` and `puts()` 125
7.2.3 `getch()` and `getche()` 126
7.2.4 `putch()` 127

7.3 Formatted Functions 128
7.3.1 `printf()` Function 128
7.3.2 Formatting with `printf()` 129
7.3.3 `scanf()` Function 134
7.3.4 Formatting with scanf 134
7.4 Review Questions 137
7.4.1 Short Answer Questions 137
7.4.2 Programming Questions 138
7.4.3 Subjective Questions 139

8. Control Structures 141
8.1 Introduction 141
8.2 Selection with if Statements 143
8.2.1 Some Points to Remember 145
8.3 if-else Statement 146
8.3.1 Write a Program to Check Whether a Number Entered by the User is Zero or Nonzero 148
8.3.2 Write a Program to Calculate the Travel Fare of a Person 149
8.4 Nested if-else Statements 150
8.4.1 Write a Program to Find the Biggest Among Three Numbers 151
8.5 if-else-if Ladders 151
8.5.1 Write a Program to Perform as a Four-Function Calculator 152
8.6 Compound Statements 154
8.7 Multiway Selection with Switch Statements 155
8.7.1 Some Points to Remember 157
8.8 goto Statement 159
8.8.1 Notes on goto 159
8.9 Introduction to Loops 160
8.10 while Loops 161
8.11 do-while Loops 164
8.11.1 Difference Between while and do-while Loops 166
8.12 for Loops 167
8.12.1 Some Solved Problems (Printing Patterns) 171
8.13 Unconditional Branching: Break and Continue 173
8.13.1 break Statements 173
8.13.2 continue Statements 174
8.14 Review Questions 176
8.14.1 Short Questions 176
8.14.2 Long Questions 176

9. Functions 179
9.1 Introduction 179
9.2 The Need for Functions 181
9.3 Types of Function 182
9.4 User-defined Functions 182
9.5 Components and Working of a Function 186
9.5.1 Calling Function 186
9.5.2 Called Function 186
9.5.3 Function Prototype 187

9.5.4 Function Definition 187
9.5.5 Function Call 188
9.5.6 Actual Arguments 188
9.5.7 Formal Arguments 188
9.5.8 Return Type 188
9.6 Categories of a Function 191
9.6.1 A Function Without Arguments and Without Return Types 191
9.6.2 A Function Without Arguments and With Return Types 191
9.6.3 A Function With Arguments and Without Return Types 193
9.6.4 A Function With Arguments and With Return Types 193
9.7 Recursion 195
9.7.1 Example: Find the Value of x^y 198
9.7.2 Programming Examples 201
9.8 Storage Classes 204
9.8.1 Automatic Storage Class 205
9.8.2 Register Storage Class 206
9.8.3 Static Storage Class 206
9.8.4 External Storage Class 207
9.9 Review Question 209
9.9.1 Objective Questions 209
9.9.2 Subjective Questions 210
9.9.3 Programming Questions 210

10. Arrays and Strings 213
10.1 Introduction 213
10.2 Need for Arrays 214
10.3 Types of Arrays 214
10.4 1D Arrays 215
10.4.1 Declaration of 1D Arrays 215
10.4.2 Initialization of Arrays 216
10.4.3 Accessing Array Elements 217
10.4.4 Characteristics of an Array 218
10.4.5 Entering Data in an Array 219
10.4.6 Displaying the Content of an Array 220
10.4.7 Programming Examples 221
10.4.7.1 Write a Program to Create an Array of N Elements and Write the Code to Find the Biggest Number and the Smallest Number Present in the Array 221
10.4.7.2 Write a Program to Search for an Element Present in the Array, the Number of Times the Element is Present, and Print the Element's Positions 222
10.4.7.3 Write a Program to Print the Binary Equivalent of a Decimal Number Using an Array 223
10.4.8 Points to Note 224
10.5 2D Arrays 225
10.5.1 Introducing Matrices 225
10.5.2 Declaration of a 2D Array 226
10.5.3 Representation of a 2D Array in Memory 226
10.5.3.1 Row Major Order 227

10.5.3.2 Column Major Order 227
10.5.4 Initialization of 2D Array 228
10.5.5 Accessing the Elements of a 2D Array 229
10.5.6 Entering Data in a 2D Array 231
10.5.7 Exploration of a 2D Matrix 234
10.5.8 Programming Examples 235
10.5.8.1 Write a Program to Add All the Elements Present in the Main Diagonal of a 2D Matrix 235
10.5.8.2 Write a Program to Add the Elements of Each Column and Print it in the Following Format 236
10.5.8.3 Write a Program to Add Two Matrices 238
10.5.8.4 Write a Program to Multiply Two Matrices 240
10.6 Multidimensional Arrays 242
10.6.1 Declaration and Representation of 3D Arrays 242
10.6.1.1 Write a Program to Declare a 3D Array, Input Some Numbers, and Display the 3D Array 244
10.7 Character Arrays: Strings 245
10.7.1 Declaration of a String 245
10.7.2 Initialization of a String 245
10.7.3 Reading a String 246
10.7.3.1 Disadvantages of the `scanf()` Function 246
10.7.3.2 Reading Strings with the `gets()` Function 247
10.7.4 Displaying the String 247
10.7.5 Programming Examples 249
10.7.5.1 Find the Length of a String 249
10.7.5.2 Count the Number of Words Present in a String 249
10.7.5.3 Reverse the String 250
10.7.5.4 Check Whether the String is a Palindrome or Not 251
10.8 String Functions 252
10.8.1 strcpy (Destination, Source) 253
10.8.2 strcat (Destination, Source) 253
10.8.3 strcmp (First, Second) 253
10.8.4 Programming Examples Using String Functions 254
10.9 Review Questions 255
10.9.1 Objective Questions 255
10.9.2 Subjective Questions 255
10.9.3 Programming Exercises 256

11. Pointers 259
11.1 Introduction 259
11.2 Basic Knowledge 260
11.3 Pointer Variables 261
11.3.1 Declaration of Pointer Variables 261
11.3.2 Working with Pointers 261
11.3.3 Workout 263
11.4 Pointer to Pointer (Double Pointer) 265
11.5 Void Pointers 266
11.6 Null Pointers 268
11.6.1 What is the Meaning of NULL? 268

11.7 Constant Pointers 268
11.7.1 Pointers to Constants 272
11.8 Pointer Arithmetic 272
11.9 Pointers and Functions 276
11.9.1 Pass by Value 276
11.9.2 Pass by Reference or Address 277
11.9.2.1 Problem: Write a Program to Swap Two Numbers Using Functions 280
11.10 Pointers and Arrays 282
11.11 Passing Arrays to Functions 285
11.11.1 Write a Program to Pass an Array to a Function and Find the Largest and Smallest Numbers Present in that Array 290
11.12 Pointers and 2D Arrays 291
11.13 Pointers and Strings 293
11.13.1 Passing a String to a Function 294
11.13.2 Write a Program to Reverse a String Using a Function 294
11.14 An Array of Pointers 295
11.15 Pointers to Functions 297
11.16 Review Questions 300
11.16.1 Objective Questions 300
11.16.2 Subjective Questions 302
11.16.3 Programming Exercises 302

12. Structures and Unions 305
12.1 Introduction 305
12.2 Declaring a Structure 307
12.2.1 Tagged Structure Declaration 307
12.2.2 Structure Declaration Using typedef 308
12.2.3 Declaring Structure Variables 308
12.2.3.1 Declaring Structure Variables Using the Structure Name 308
12.2.3.2 Declaring Structure Variables after the Closing Braces 309
12.3 Initializing a Structure 311
12.4 Accessing Structure Members 312
12.4.1 Accessing Members Using the dot (.) Operator 313
12.5 Learn to Code Examples 315
12.6 Arrays of Structures 316
12.7 Structures within Structures (Nested Structures) 318
12.7.1 Declaration of Nested Structures 318
12.7.1.1 Declare the Structure with One Declaration 319
12.7.1.2 Declare the Structure Separately 319
12.7.2 Accessing the Members of a Nested Structure 319
12.7.3 Nested Structure Initialization 320
12.8 User-defined Data Type: *typedef* 321
12.8.1 Uses of typedef 322
12.9 Pointers and Structures 323
12.9.1 Accessing Structure Members Using a Pointer 323
12.9.2 A Pointer as a Member of a Structure 324
12.9.3 Self-referential Structures 324
12.10 Structures and Functions 328

12.10.1 Passing Individual Members of a Structure 328
12.10.2 Passing the Whole Structure Using the Pass by Value Concept 330
12.10.3 Passing the Whole Structure Using the Pass by Address Concept 332
12.11 Unions 333
12.11.1 Declaration of a Union 333
12.11.2 Member Accessing 336
12.12 Structures vs. Unions 336
12.12.1 Size of Unions and Structures 337
12.12.2 Sharing Memory and Member Accessing 337
12.13 Bitfields 338
12.13.1 Declaration of a Bitfield 340
12.13.2 Uses of Bitfields 342
12.14 Enumeration 343
12.15 Review Questions 346
12.15.1 Objective Questions 346
12.15.2 Subjective Questions 346
12.15.3 Programming Exercises 346

13. Dynamic Memory Allocation 349
13.1 Introduction 349
13.1.1 Process of Memory Allocation 350
13.1.1.1 Text Segments 351
13.1.1.2 Data Segments 351
13.1.1.3 Stack Segments 351
13.1.1.4 Heap Segments 351
13.2 Types of Memory Allocation 351
13.2.1 Static Memory Allocation 351
13.2.2 Dynamic Memory Allocation 351
13.3 Dynamic Memory Allocation Process 352
13.3.1 The `malloc()` Function 353
13.3.2 The `calloc()` Function 356
13.3.3 The `realloc()` Function 356
13.3.4 The `free()` Function 360
13.4 Review Questions 361

14. File Handling 365
14.1 Introduction 365
14.1.1 Difference between Console I/O and File I/O 366
14.2 Basics of File I/O 367
14.2.1 What is a File? 367
14.2.2 File Handling Process Flow 367
14.3 Opening a File 368
14.4 Closing a File 370
14.5 File Functions with Examples 371
14.5.1 The `fprintf()` and `fscanf()` Functions 371
14.5.1.1 Writing and Reading an Integer Using `fprintf()` and `fscanf()` 372
14.5.2 The `putw()` and `getw()` Functions 374

14.5.2.1 Writing and Reading More than One Integer Using the `putw()` and `getw()` Functions 374
14.5.2.2 Reading Numbers from a File and Checking Them for Even or Odd 375
14.5.3 The `fputc()` and `fgetc()` Functions 377
14.5.3.1 Writing and Reading a Character Using `fputc()` and `fgetc()` 378
14.5.3.2 Writing and Reading Multiple Characters Using `fputc()` and `fgetc()` 379
14.5.3.3 Count Number of Characters, Lines, Tabs, and Blank Spaces Present in a File 380
14.5.4 The `fputs()` and `fgets()` Functions 381
14.5.4.1 Writing and Reading a String Using `fputs()` and `fgets()` 382
14.6 Other Programming Examples 383
14.7 Review Questions 386

15. The Preprocessor 389
15.1 Introduction 389
15.2 Preprocessor Directives 389
15.3 Macro-substitutions 390
15.3.1 Writing Macros with Arguments 392
15.3.2 Removing a Macro 392
15.4 The #include Preprocessor 392
15.5 Conditional Preprocessors 395
15.5.1 The #ifdef and #endif Preprocessor Directives 395
15.5.2 The #ifndef and #endif Directives 396
15.5.3 The #if and #endif Directives 396
15.6 Other Preprocessor Directives 396
15.6.1 #line Directives 397
15.6.2 #error Directives 398
15.6.3 #pragma Directives 400
15.7 Review Questions 400

16. Command Line Arguments 403
16.1 Introduction 403
16.1.1 The Code::Block IDE 404
16.2 Executing a Program Using a Command Prompt 405
16.2.1 Installing the minGW Compiler 405
16.2.2 Compiling and Executing a Program 407
16.3 Fundamentals of the Command Line Argument 410
16.4 Using Command Line Arguments 411
16.5 Review Questions 413

Appendix A: ASCII Character Table 415
Appendix B: Integer Representation 417
Index 423