
Contents

List of Figures	xiii
List of Tables	xvii
Foreword for the First Edition	xix
Recommendations for the First Edition	xxi
Preface	xxiii
Acknowledgments	xxvii
Source Code	xxix
Author and Artist	xxxi
I Storage: Memory and File	1
1 Program Execution	3
1.1 Compile	3
1.2 Redirect Output	5
1.3 Problems	6
1.3.1 Replace String	6
1.3.2 Keep Third Column	6
1.3.3 Find String in a File	6
1.3.4 Sort by a Column	6
1.3.5 gcc Warnings	6
1.3.6 du command	7
1.3.7 Show End of a File	7
2 Stack Memory	8
2.1 Values and Addresses	8
2.2 Stack	9
2.3 The Call Stack	10
2.3.1 The Return Location	10
2.3.2 Function Arguments	12
2.3.3 Local Variables	14
2.3.4 Value Address	14
2.3.5 Arrays	15
2.3.6 Retrieving Addresses	16
2.4 Visibility	17
2.5 Examine the Call Stack with DDD	19
2.6 Problems	22

2.6.1	Draw Call Stack I	22
2.6.2	Draw Call Stack II	23
2.6.3	Addresses	23
Prevent, Detect, and Remove Bugs		24
3.1	Rules in Software Development	24
3.2	Developing Software $\neq$ Coding	26
3.2.1	Before Coding	26
3.2.2	During Coding	27
3.2.3	After Coding	28
3.3	Post-Execution and Interactive Debugging	28
3.4	Separate Testing Code from Production Code	29
3.5	Use <code>assert</code> Correctly	30
3.6	How to Comment Code	30
Pointers		32
4.1	Scope	32
4.2	The Swap Function	33
4.3	Pointers	35
4.4	Self Test: Pointers	37
4.5	The Swap Function Revisited	38
4.6	Type Errors	40
4.7	Arrays and Pointers	41
4.8	Type Rules	44
4.9	Pointer Arithmetic	45
4.10	Sizes of Data Types	46
4.11	Problems	49
4.11.1	Swap Function 1	49
4.11.2	Swap Function 2	49
4.11.3	Swap Function 3	50
4.11.4	Swap Function 4	50
4.11.5	Swap Function 5	51
4.11.6	15,552 Variations	51
4.11.7	Pointer and Array	53
Writing and Testing Programs		54
5.1	Distinct Array Elements	54
5.1.1	<code>main</code> Function	54
5.1.2	<code>areDistinct</code> Function	56
5.1.3	Compiling and Linking	57
5.2	Build and Test Program with Multiple Files	58
5.2.1	<code>make</code> and Makefile	58
5.2.2	Test using <code>make</code>	61
5.2.3	Generate Test Cases	62
5.3	Detect Invalid Memory Access	63
5.4	Test Coverage	65
5.5	Limit Core Size	67
5.6	Remove Large Files	68
5.7	Problems	68
5.7.1	Array Index	68

5.7.2	<code>gcov</code>	68
5.7.3	Test Coverage	68
6 Strings		69
6.1	Array of Characters	69
6.2	String Functions in C	71
6.2.1	Copy: <code>strcpy</code>	71
6.2.2	Copy: <code>strdup</code>	72
6.2.3	Compare: <code>strcmp</code>	72
6.2.4	Find Substrings: <code>strstr</code>	73
6.2.5	Find Characters: <code>strchr</code>	73
6.3	Understand <code>argv</code>	73
6.4	Count Substrings	74
6.5	Problems	76
6.5.1	Compare Strings	76
6.5.2	Count Substring-1	76
6.5.3	Count Substring-2	76
7 Heap Memory		77
7.1	Allocate Memory using <code>malloc</code>	77
7.2	Stack and Heap Memory	79
7.3	Functions that Return a Heap Address	81
7.4	Two-Dimensional Arrays in C	82
7.5	Pointers and Function Arguments	85
7.6	Problems	87
7.6.1	<code>free</code>	87
7.6.2	<code>malloc</code>	87
7.6.3	Address of Stack Memory	87
8 Programming Problems Using Heap Memory		88
8.1	Sort an Array	88
8.1.1	Generate Test Input and Expected Output	88
8.1.2	Sort Integers	88
8.1.3	Makefile	91
8.1.4	Use <code>valgrind</code> to Detect Memory Leaks	92
8.1.5	Use <code>gcc -fsanitize</code> to Detect Memory Leaks	93
8.2	Sort Using <code>qsort</code>	94
8.2.1	<code>qsort</code>	94
8.2.2	The Comparison Function	95
8.2.3	Execution Examples	96
8.2.4	Sort Strings	97
8.3	Problems	99
8.3.1	Bubble Sort	99
8.3.2	Selection Sort	100
8.3.3	Comparison in <code>qsort</code>	100
9 Reading and Writing Files		101
9.1	Read from Files	101
9.2	Write to Files	104
9.3	Read and Write Strings	106

Programming Problems Using File	110
1.1 Sort a File of Integers	110
1.2 Count the Occurrences of Characters	112
1.3 Count the Occurrences of a Word	114
1.4 Problems	115
10.4.1 Determine Palindrome	115
10.4.2 Compare Two Files	115
10.4.3 fscanf and ftell	115
Array Index, Security, and Trends	117
1.1 Array Index	117
1.2 Modify Program Behavior	120
1.3 Trends in Programming Languages	121
Version Control	123
2.1 Version Control Overview	123
2.2 git and github.com	123
2.3 Create a Repository on github	124
2.4 Sample Repository	128
2.5 Use git and github.com	131
2.6 Problems	135
12.6.1 git commit and git push	135
12.6.2 Revert to an earlier version	135
12.6.3 When to commit?	136
12.6.4 When to clone?	136
12.6.5 Recover deleted file	136
Recursion	137
Concept	139
13.1 Select Balls with Restrictions	139
13.1.1 Balls of Two Colors	139
13.1.2 Balls of Three Colors	140
13.2 One-Way Streets	141
13.3 The Tower of Hanoi	142
13.4 Calculate Integer Partitions	143
13.4.1 Count the Number of "1"s	144
13.4.2 Odd Numbers Only	145
13.4.3 Increasing Values	146
13.4.4 Alternating Odd and Even Numbers	146
13.5 Problems	148
13.5.1 Parentheses	148
13.5.2 Ball Selection	149
13.5.3 Partition using Odd and Even Numbers	149
Recursive C Functions	150
14.1 Select Balls with Restrictions	150
14.2 One-Way Streets	151
14.3 The Tower of Hanoi	152
14.4 Integer Partition	154
14.5 Factorial	155

14.6 Fibonacci Numbers	157
14.7 Performance Profiling with gprof	161
14.8 Problems	162
14.8.1 Count Number of Calls: Original Definition	162
14.8.2 Count Number of Calls: Efficient Recursion	162
14.8.3 Parentheses	163
14.8.4 Fibonacci Numbers	164
15 Integer Partition	165
15.1 Print Partitions	165
15.2 Stack and Heap Memory	166
15.3 Trace Recursive Function Calls	168
15.4 Generate Partitions with Restrictions	170
15.4.1 Using Odd Numbers Only	172
15.4.2 Using Sequences of Increasing Numbers	172
15.4.3 Using Alternating Odd and Even Numbers	173
15.5 Problems	174
15.5.1 Even Numbers Only	174
15.5.2 Decreasing Values	174
15.5.3 Error in partition	174
15.5.4 Error in partition	174
15.5.5 Error in partition	174
16 Programming Problems Using Recursion	175
16.1 Binary Search	175
16.2 Quick Sort	177
16.3 Permutations and Combinations	182
16.4 Stack Sort	185
16.5 An Incorrect Recursive Function	189
16.6 Problems	190
16.6.1 Shuffle Cards - 1	190
16.6.2 Shuffle Cards - 2	191
16.6.3 Shuffle Cards - 3	191
III Structure	193
17 Programmer-Defined Data Types	195
17.1 Struct and Object	195
17.2 Objects as Arguments	199
17.3 Objects and Pointers	200
17.4 Constructors and Destructors	204
17.5 Structures within Structures	211
17.6 Binary Files and Objects	213
17.7 Problems	216
17.7.1 Structure and Pointer	216
17.7.2 Structure and Attributes' Sizes	217
17.7.3 Alternative Ways to Assign Attributes	219

Programming Problems Using Structure	221
18.1 Sort a Person Database	221
18.2 Packing Decimal Digits	226
18.3 Binary File and Pointer	234
18.4 Problems	236
18.4.1 Number Systems	236
18.4.2 Structure with Pointer-1	236
18.4.3 Structure with Pointer-2	236
Linked Lists	237
19.1 Dynamic Data Structure	237
19.2 Linked Lists	238
19.3 Insert Data	239
19.4 Search a Linked List	240
19.5 Delete a Node from a Linked List	241
19.6 Print a Linked List	243
19.7 Destroy a Linked List	244
Programming Problems Using Linked List	246
20.1 Queues	246
20.2 Sort Numbers	247
20.3 Sparse Arrays	248
20.4 Reversing a Linked List	252
20.5 Doubly Linked List	254
Binary Search Trees	255
21.1 Binary Tree	255
21.2 Binary Search Tree	256
21.3 Insert Data into a Binary Search Tree	257
21.4 Search a Binary Search Tree	259
21.5 Print a Binary Tree	260
21.6 Delete from a Binary Search Tree	262
21.7 Destroy a Binary Tree	264
21.8 Count the Different Shapes of a Binary Tree	264
21.9 Problems	266
21.9.1 Count Nodes	266
21.9.2 Tree Height	266
21.9.3 Check Binary Search Tree	266
21.9.4 Ancestor-Offspring	266
21.9.5 Build Binary Tree	266
Parallel Programming Using Threads	268
22.1 Parallel Programming	268
22.2 POSIX Threads	269
22.3 Subset Sum	270
22.3.1 Sequential Solution	272
22.3.2 Multiple-Threads Solution	275
22.4 Interleave the Execution of Threads	279
22.5 Thread Synchronization	282
22.6 Amdahl's Law	285
22.6.1 Speedup vs. Efficiency	286

22.6.2 Understanding Amdahl's Law	287
22.6.3 The Limit's of Amdahl's Law	288
22.7 Problems	288
22.7.1 Interleaving of Threads	288
22.7.2 Interleaving of Threads	288
23 Unit Test	289
23.1 Google Test Framework	289
23.2 Rational Numbers	292
23.3 Rational Numbers and Operations	295
23.3.1 Create Ratoinal Object	295
23.3.2 Arithmetic Operations	297
23.4 Use and Test Rational Numbers	300
23.4.1 Use Rational Numbers	300
23.4.2 Test Rational Numbers	303
23.5 Testing Strategies	308
IV Applications	311
24 Find the Exit of a Maze	313
24.1 Maze File Format	313
24.2 Strategy to Find Exit	314
24.3 Implement the Strategy	316
24.4 Problems	321
24.4.1 Mark of Bricks	321
24.4.2 Multiple Exits	321
24.4.3 Order of Directions	321
24.4.4 Change if Conditions	321
25 Sudoku	322
25.1 Sudoku Game	322
25.2 Recursive Solution	323
25.3 Implement the Solution	325
25.4 Design Decisions	333
25.5 Network Communication	335
25.5.1 Server	336
25.5.2 Client	339
25.6 Problems	341
25.6.1 Check Sudoku	341
25.6.2 Transfer Large File	341
26 Image Processing	342
26.1 Structure for Image	342
26.2 Image Pixels and Colors	347
26.3 Color Filter	348
26.4 Invert Colors	349
26.5 Detect Edges	350
26.6 Equalize Colors	352
26.7 main Function	353
26.8 Problems	355
26.8.1 Color Histogram	355

26.8.2	Color Space	355
26.8.3	Gradient	355
26.8.4	Vertical Mirror	355
27	Huffman Compression	356
27.1	Variable-Length Coding	356
27.2	Compress	357
27.2.1	Count and Sort Occurrences	357
27.2.2	Build the Code Tree	359
27.2.3	Create Code Book and Compress Data	361
27.2.4	Describe the Coding Tree	361
27.2.5	Use Bits	362
27.3	Decompress	365
27.4	Implementation	366
27.4.1	Test Case Generator	366
27.4.2	main Function	367
27.4.3	Makefile	369
27.4.4	Compression	370
27.4.5	Decompression	377
27.4.6	Compression Ratios	381
27.5	Problems	381
27.5.1	Compression Tree	381
27.5.2	Compressed File	381
27.5.3	Compressed File	382
27.5.4	One-Letter Input	382
27.5.5	Compressed File	382
27.5.6	Number of Leaf Nodes	382
27.5.7	Compression Ratio	382
27.5.8	Tree Description	382
27.5.9	Highest Possible Compression Ratio	382
27.5.10	Lowest Possible Compression Ratio	382
	Index	383
	Epilogue: The Computer Engineer as Tool-User	395