

THINK LIKE A PROGRAMMER

**An Introduction to
reactive Problem Solvin**

by V. Anton Spraul



**no starch
press**

San Francisco

CONTENTS IN DETAIL

ACKNOWLEDGMENTS	xi
------------------------	-----------

INTRODUCTION	xiii
---------------------	-------------

About This Book	xv
Prerequisites	xv
Chosen Topics	xv
Programming Style	xvi
Exercises	xvi
Why C++?	xvii

1

STRATEGIES FOR PROBLEM SOLVING	1
---------------------------------------	----------

Classic Puzzles	2
The Fox, the Goose, and the Corn	3
Problem: How to Cross the River?	3
Sliding Tile Puzzles	7
Problem: The Sliding Eight	7
Problem: The Sliding Five	8
Sudoku	11
Problem: Completing a Sudoku Square	11
The Quarrasi Lock	13
Problem: Opening the Alien Lock	13
General Problem-Solving Techniques	15
Always Have a Plan	16
Restate the Problem	17
Divide the Problem	17
Start with What You Know	18
Reduce the Problem	19
Look for Analogies	20
Experiment	20
Don't Get Frustrated	21
Exercises	22

2

PURE PUZZLES	25
---------------------	-----------

Review of C++ Used in This Chapter	26
Output Patterns	26
Problem: Half of a Square	26
Problem: A Square (Half of a Square Reduction)	27
Problem: A Line (Half of a Square Further Reduction)	27
Problem: Count Down by Counting Up	28
Problem: A Sideways Triangle	29
Input Processing	31
Problem: Luhn Checksum Validation	31
Breaking Down the Problem	33

Problem: Convert Character Digit to Integer	35
Problem: Lohn Checksum Validation, Fixed Length	36
Problem: Simple Checksum Validation, Fixed Length	36
Problem: Positive or Negative	39
Putting the Pieces Together	39
Tracking State	41
Problem: Decode a Message	41
Problem: Reading a Number with Three or Four Digits	45
Problem: Reading a Number with Three or Four Digits, Further Simplified	46
Conclusion	53
Exercises	53

3 SOLVING PROBLEMS WITH ARRAYS **55**

Review of Array Fundamentals	56
Store	56
Copy	57
Retrieval and Search	57
Sort	59
Compute Statistics	61
Solving Problems with Arrays	62
Problem: Finding the Mode	62
Refactoring	65
Arrays of Fixed Data	67
Non-scalar Arrays	69
Multidimensional Arrays	71
Deciding When to Use Arrays	74
Exercises	78

4 SOLVING PROBLEMS WITH POINTERS AND DYNAMIC MEMORY **81**

Review of Pointer Fundamentals	82
Benefits of Pointers	83
Runtime-Sized Data Structures	83
Resizable Data Structures	83
Memory Sharing	84
When to Use Pointers	84
Memory Matters	85
The Stack and the Heap	86
Memory Size	88
Lifetime	90
Solving Pointer Problems	91
Variable-Length Strings	91
Problem: Variable-Length String Manipulation	91
Linked Lists	101
Problem: Tracking an Unknown Quantity of Student Records	101
Conclusion and Next Steps	108
Exercises	109

5 SOLVING PROBLEMS WITH CLASSES

11 1

Review of Class Fundamentals	112
Goals of Class Use	113
Encapsulation	114
Code Reuse	114
Dividing the Problem	1 15
Information Hiding	115
Readability	117
Expressiveness	117
Building a Simple Class	118
Problem: Class Rosier	118
The Basic Class Framework	119
Support Methods	122
Classes with Dynamic Data	125
Problem: Tracking an Unknown Quantity of Student Records	126
Adding a Node	128
Rearranging the List	130
Destructor	133
Deep Copy	134
The Big Picture for Classes with Dynamic Memory	1 39
Mistakes to Avoid	140
The Fake Class	140
Single-Taskers	141
Exercises	141

6 SOLVING PROBLEMS WITH RECURSION

143

Review of Recursion Fundamentals	144
Head and Tail Recursion	144
Problem: How Many Parrots?	144
Approach 1	145
Approach 2	146
Problem: Who's Our Best Customer?	148
Approach 1	149
Approach 2	151
The Big Recursive Idea	152
Problem: Computing the Sum of an Array of Integers	153
Common Mistakes	155
Too Many Parameters	155
Global Variables	156
Applying Recursion to Dynamic Data Structures	158
Recursion and Linked Lists	158
Problem: Counting Negative Numbers in a Singly Linked List	159
Recursion and Binary Trees	160
Problem: Find the Largest Value in a Binary Tree	162
Wrapper Functions	163
Problem: Find the Number of Leaves in a Binary Tree	163
When to Choose Recursion	165
Arguments Against Recursion	166

Problem: Display a Linked List in Order

168

Problem: Display a Linked List in Reverse Order

168

Exercises

170

7

SOLVING PROBLEMS WITH CODE REUSE

171

Good Reuse and Bad Reuse

172

Review of Component Fundamentals

173

Code Block

173

Algorithms

173

Patterns

174

Abstract Data Types

175

Libraries

175

Building Component Knowledge

176

Exploratory Learning

176

Problem: The First Student

177

As-Needed Learning

1 80

Problem: Efficient Traversal

180

Choosing a Component Type

188

Component Choice in Action

1 89

Problem: Sorting Some, Leaving Others Alone

189

Comparing the Results

193

Exercises

193

8

THINKING LIKE A PROGRAMMER

195

Creating Your Own Master Plan

196

Playing to Your Strengths and Weaknesses

1 96

Putting the Master Plan Together

202

Tackling Any Problem

203

Problem: Cheating at Hangman

204

Finding a Way to Cheat

205

Required Operations for Cheating at Hangman

206

Initial Design

208

Initial Coding

210

Analysis of Initial Results

217

The Art of Problem Solving

218

Learning New Programming Skills

219

New Languages

219

New Skills for a Language You Already Know

222

New Libraries

223

Take a Class

223

Conclusion

224

Exercises

225

INDEX

227