

Index of Programs

Elements of Programming

Your First Program

- 1.1.1 Hello, World 6
- 1.1.2 Using a command-line argument 8

Built-in Types of Data

- 1.2.1 String concatenation example. 20
- 1.2.2 Integer multiplication and division 22
- 1.2.3 Quadratic formula 24
- 1.2.4 Leap year 27
- 1.2.5 Casting to get a random integer 33

Conditionals and Loops

- 1.3.1 Flipping a fair coin 49
- 1.3.2 Your first while loop. 51
- 1.3.3 Computing powers of two 53
- 1.3.4 Your first nested loops 59
- 1.3.5 Harmonic numbers 61
- 1.3.6 Newton's method. 62
- 1.3.7 Converting to binary 64
- 1.3.8 Gambler's ruin simulation 66
- 1.3.9 Factoring integers. 69

Arrays

- 1.4.1 Sampling without replacement 94
- 1.4.2 Coupon collector simulation 98
- 1.4.3 Sieve of Eratosthenes 100
- 1.4.4 Self-avoiding random walks 109

Input and Output

- 1.5.1 Generating a random sequence 122
- 1.5.2 Interactive user input 129
- 1.5.3 Averaging a stream of numbers 130
- 1.5.4 A simple filter 134
- 1.5.5 Input-to-drawing filter 139
- 1.5.6 Bouncing ball 145
- 1.5.7 Digital signal processing 150

Case Study: Random Web Surfer

- 1.6.1 Computing the transition matrix 163
- 1.6.2 Simulating a random surfer 165
- 1.6.3 Mixing a Markov chain 172

Functions and Modules

Static Methods

- 2.1.1 Newton's method (revisited) 184
- 2.1.2 Gaussian functions 193
- 2.1.3 Coupon collector (revisited) 195
- 2.1.4 Play that Tune (revisited) 203

Libraries and Clients

- 2.2.1 Random number library 224
- 2.2.2 Array IO library 228
- 2.2.3 Iterated function systems. 231
- 2.2.4 Data analysis library. 235
- 2.2.5 Plotting data values in an array 237
- 2.2.6 Bernoulli trials 240

Recursion

- 2.3.1 Euclid's algorithm. 257
- 2.3.2 Towers of Hanoi 260
- 2.3.3 Gray code 265
- 2.3.4 Recursive graphics 267
- 2.3.5 Brownian bridge 269

Case Study: Percolation

- 2.4.1 Percolation scaffolding. 288
- 2.4.2 Vertical percolation detection. 290
- 2.4.3 Visualization client 293
- 2.4.4 Percolation probability estimate. 295
- 2.4.5 Percolation detection 297
- 2.4.6 Adaptive plot client 300

Object-Oriented Programming

Data Types

3.1.1	Charged particles	318
3.1.2	Albers squares	323
3.1.3	Luminance library	326
3.1.4	Converting color to grayscale . .	329
3.1.5	Image scaling	331
3.1.6	Fade effect	333
3.1.7	Visualizing electric potential . .	335
3.1.8	Finding genes in a genome . . .	340
3.1.9	Concatenating files	345
3.1.10	Screen scraping for stock quotes	347
3.1.11	Splitting a file	348

Creating Data Types

3.2.1	Charged-particle implementation	373
3.2.2	Stopwatch	377
3.2.3	Histogram	379
3.2.4	Turtle graphics	382
3.2.5	Spira mirabilis	385
3.2.6	Complex numbers	391
3.2.7	Mandelbrot set	395
3.2.8	Stock account	399

Designing Data Types

3.3.1	Complex numbers (alternate). .	419
3.3.2	Counter	423
3.3.3	Spatial vector.	430
3.3.4	Document.	439
3.3.5	Similarity detection	442

Case Study: N-body Simulation

3.4.1	Gravitational body	460
3.4.2	N-body simulation	463

Algorithms and Data Structures

Performance

4.1.1	3-sum problem	475
4.1.2	Validating a doubling hypothesis	477

Sorting and Searching

4.2.1	Binary search (20 questions) . .	512
4.2.2	Bisection search	515
4.2.3	Binary search (sorted array) . .	517
4.2.4	Insertion sort	524
4.2.5	Doubling test for insertion sort .	526
4.2.6	Mergesort	528
4.2.7	Frequency counts	533
4.2.8	Longest repeated substring . . .	539

Stacks and Queues

4.3.1	Stack of strings (array).	554
4.3.2	Stack of strings (linked list). . .	559
4.3.3	Stack of strings (array doubling)	563
4.3.4	Generic stack.	568
4.3.5	Expression evaluation	572
4.3.6	Generic FIFO queue (linked list)	578
4.3.7	M/D/1 queue simulation.	583
4.3.8	Load balancing simulation	591

Symbol Tables

4.4.1	Dictionary lookup	614
4.4.2	Indexing.	617
4.4.3	BST symbol table	625
4.4.4	Dedup filter	633

Case Study: Small World

4.5.1	Graph data type	657
4.5.2	Using a graph to invert an index.	661
4.5.3	Shortest-paths client	665
4.5.4	Shortest-paths implementation .	670
4.5.5	Small-world test	677