



4. ALGORITHMS AND DATA STRUCTURES

Overview. In this chapter we describe and implement some of the most important algorithms and data structures in use on computers today. (For a more in-depth treatment, we recommend the companion textbook [Algorithms, 4th Edition](#).) We begin by considering a powerful framework for measuring and analyzing the efficiency of our programs. This enables us to compare algorithms and accurately predict performance. Next, we consider several novel algorithms for the classic problem of sorting. Then, we build the most important higher level data structures, including stacks, queues, and symbol tables.

- [4.1 Performance](#) outlines a scientific method and powerful theory for understanding the performance and resource consumption of the program that we write.
- [4.2 Sorting and Searching](#) describes two classical algorithms—mergesort and binary search—along with several applications where their efficiency plays a critical role.
- [4.3 Stacks and Queues](#) introduces two closely related data structures for manipulating arbitrary large collections of data.
- [4.4 Symbol Tables](#) considers a quintessential data structure known as the symbol table for storing information, and two efficient implementations—hash tables and binary search trees.
- [4.5 Small World Phenomenon](#) presents a case study to investigate the small world phenomenon—the principle that we are all linked by short chains of acquaintances.

Java programs in this chapter. Below is a list of Java programs in this chapter. Click on the program name to access the Java code; click on the reference number for a brief description; read the textbook for a full discussion.

REF	PROGRAM	DESCRIPTION
4.1.1	ThreeSum.java	3-sum problem
4.1.2	DoublingTest.java	validating a doubling hypothesis
4.2.1	Questions.java	binary search (20 questions)
4.2.2	Gaussian.java	bisection search
4.2.3	BinarySearch.java	binary search (in a sorted array)
4.2.4	Insertion.java	insertion sort
4.2.5	InsertionTest.java	doubling test for insertion sort
4.2.6	Merge.java	mergesort
4.2.7	FrequencyCount.java	frequency counts
4.3.1	ArrayStackOfStrings.java	stack of strings (array)
4.3.2	LinkedStackOfStrings.java	stack of strings (linked list)
4.3.3	ResizingArrayStackOfStrings.java	stack of strings (resizing array)
4.3.4	Stack.java	generic stack
4.3.5	Evaluate.java	expression evaluation
4.3.6	Queue.java	generic queue
4.3.7	MM1Queue.java	M/M/1 queue simulation
4.3.8	LoadBalance.java	load balancing simulation
4.4.1	Lookup.java	dictionary lookup
4.4.2	Index.java	indexing
4.4.3	HashST.java	hash table
4.4.4	BST.java	binary search tree
4.4.5	DeDup.java	dedup filter
4.5.1	Graph.java	graph data type
4.5.2	IndexGraph.java	using a graph to invert an index
4.5.3	PathFinder.java	shortest-paths client
4.5.4	PathFinder.java	shortest-paths implementation
4.5.5	SmallWorld.java	small-world test
4.5.6	Performer.java	performer–performer graph