Exercises 8.2-1

Using Figure 8.2 as a model, illustrate the operation of COUNTING-SORT on the array A =

­6, 0, 2, 0, 1, 3, 4, 6, 1, 3, 2­.

Exercises 8.2-3

Suppose that the for loop header in line 9 of the COUNTING-SORT procedure is rewritten as

for j ← 1 to length[A]

Show that the algorithm still works properly. Is the modified algorithm stable?

Exercises 8.3-1

Using Figure 8.3 as a model, illustrate the operation of RADIX-SORT on the following list ofEnglish words: COW, DOG, SEA, RUG, ROW, MOB, BOX, TAB, BAR, EAR, TAR, DIG,BIG, TEA, NOW, FOX.

Exercises 8.4-1

Using Figure 8.4 as a model, illustrate the operation of BUCKET-SORT on the array A =

­.79, .13, .16, .64, .39, .20, .89, .53, .71, .42­.

Exercises 9.2-4

Suppose we use RANDOMIZED-SELECT to select the minimum element of the array A = ­3, 2, 9, 0, 7, 5, 4, 8, 6, 1­. Describe a sequence of partitions that results in a worst-case performance of RANDOMIZED-SELECT.

Exercises 9.3-7

Describe an O(n)-time algorithm that, given a set S of n distinct numbers and a positive integer k ≤ n, determines the k numbers in S that are closest to the median of S.

Exercises 11.2-2

Demonstrate the insertion of the keys 5, 28, 19, 15, 20, 33, 12, 17, 10 into a hash table with collisions resolved by chaining. Let the table have 9 slots, and let the hash function be h(k) = k mod 9.

Exercises 11.3-4

Consider a hash table of size m = 1000 and a corresponding hash function h(k) = ⌊m(k A mod 1)⌋ for A= (sqrt(5)-1)/2. Compute the locations to which the keys 61, 62, 63, 64, and 65 are mapped.

Exercises 11.4-1

Consider inserting the keys 10, 22, 31, 4, 15, 28, 17, 88, 59 into a hash table of length m = 11 using open addressing with the primary hash function h'(k) = k mod m. Illustrate the result of inserting these keys using linear probing, using quadratic probing with c1 = 1 and c2 = 3, and using double hashing with h2(k) = 1 + (k mod (m - 1)).

Exercises 12.1-1

For the set of keys {1, 4, 5, 10, 16, 17, 21}, draw binary search trees of height 2, 3, 4, 5, and 6.

Exercises 12.2-1

Suppose that we have numbers between 1 and 1000 in a binary search tree and want to search for the number 363. Which of the following sequences could not be the sequence of nodes examined?

a. 2, 252, 401, 398, 330, 344, 397, 363.

b. 924, 220, 911, 244, 898, 258, 362, 363.

c. 925, 202, 911, 240, 912, 245, 363.

d. 2, 399, 387, 219, 266, 382, 381, 278, 363.

e. 935, 278, 347, 621, 299, 392, 358, 363.

Exercises 12.3-5

Is the operation of deletion "commutative" in the sense that deleting x and then y from a binary search tree leaves the same tree as deleting y and then x? Argue why it is or give a counterexample.

Exercises 13.3-2

Show the red-black trees that result after successively inserting the keys 41, 38, 31, 12, 19, 8 into an initially empty red-black tree.

Exercises 13.3-5

Consider a red-black tree formed by inserting n nodes with RB-INSERT. Argue that if n > 1, the tree has at least one red node.

Exercises 13.4-3

In Exercise 13.3-2, you found the red-black tree that results from successively inserting the keys 41, 38, 31, 12, 19, 8 into an initially empty tree. Now show the red-black trees that result from the successive deletion of the keys in the order 8, 12, 19, 31, 38, 41.

Exercises 14.1-1

Show how OS-SELECT(T, 10) operates on the red-black tree T of Figure 14.1.

Exercises 15.2-1

Find an optimal parenthesization of a matrix-chain product whose sequence of dimensions is

­5, 10, 3, 12, 5, 50, 6­.

Exercises 15.4-1

Determine an LCS of ­1, 0, 0, 1, 0, 1, 0, 1­ and ­0, 1, 0, 1, 1, 0, 1, 1, 0­.

Exercises 16.1-3

Suppose that we have a set of activities to schedule among a large number of lecture halls. We wish to schedule all the activities using as few lecture halls as possible. Give an efficient greedy algorithm to determine which activity should use which lecture hall.

Exercises 16.2-1

Prove that the fractional knapsack problem has the greedy-choice property.

Exercises 16.3-2

What is an optimal Huffman code for the following set of frequencies a:1 b:1 c:2 d:3 e:5 f:8 g:13 h:21?

Problems 16-1: Coin changing

Consider the problem of making change for n cents using the fewest number of coins. Assume that each coin's value is an integer.

a. Describe a greedy algorithm to make change consisting of quarters, dimes, nickels, and pennies. Prove that your algorithm yields an optimal solution.

b. Suppose that the available coins are in the denominations that are powers of c, i.e., the denominations are c0, c1, ..., ck for some integers c >; 1 and k ≥ 1. Show that the greedy algorithm always yields an optimal solution.

c. Give a set of coin denominations for which the greedy algorithm does not yield an optimal solution. Your set should include a penny so that there is a solution for every value of n.