HW 11 textbook problems and hints

5.6 (1 - 20)

10. ▷a. *Moldy chocolate* Two payers take turns by breaking an m-by-n chocolate bar, which has one spoiled 1-by-1 square. Each break must be a single straight line cutting all the way across the bar along the boundaries between the squares. After each break, the player who broke the bar last eats the piece that does not contain the spoiled corner. The player left with the spoiled square loses the game. Is it better to go first or second in this game?

My instructions from the assignment document: This problem may be harder than it looks at first. Perhaps it is no accident that the problem is in section 5.6, and also very close to Chapter 6. "Transform and conquer" is a good way to find a complete solution. However,
If you can't solve the general case, solve some cases that you can solve, and talk about what you tried for other cases.

Author's hint:

> Play several rounds of the game on the graphed paper to become comfortable with the problem. Considering special cases of the spoiled square's location should help you to solve it.

6.1 (2 - 10)

7. To sort or not to sort? Design a reasonably efficient algorithm for solving each of the following problems and determine its efficiency class.

   a. You are given n telephone bills and m checks sent to pay the bills ($n \geq m$). Assuming that telephone numbers are written on the checks, find out who failed to pay. (For simplicity, you may also assume that only one check is written for a particular bill and that it covers the bill in full.)

   b. You have a file of n student records indicating each student's number, name, home address, and date of birth. Find out the number of students from each of the 50 U.S. states.

Author's hint:

> a. The problem is similar to one of the preceding problems in these exercises.
>
> b. How would you solve this problem if the student information were written on index cards? Better yet, think how somebody else, who has never taken a course on algorithms but possesses a good dose of common sense, would solve this problem.

6.2 (3 - 10) problem 8c

The ***Gauss-Jordan elimination*** method differs from Gaussian elimination in that the elements above the main diagonal of the coefficient matrix are made zero at the same time and by the same use of a pivot row as the elements below the main diagonal.

a. Apply the Gauss-Jordan method to the system of Problem 1 of these exercises.

b. What general design technique is this algorithm based on?

c.▷ In general, how many multiplications are made by this method while solving a system of n equations in n unknowns? How does this compare with the number of multiplications made by the Gaussian elimination method in both its elimination and its back-substitution stages?

Students are only required to do part c. I put the rest here for context.

Author's hint:

8. a. Manipulate the matrix rows above a pivot row the same way the rows below the pivot row are changed.

   b. Are the Gauss-Jordan method and Gaussian elimination based on the same algorithm design technique or on different ones?

   c. Derive the formula for the number of multiplications in the Gauss-Jordan method the same way it was done for Gaussian elimination in Section 6.2.

6.3 (4 - 6)

7. a. Construct a 2-3 tree for the list C, O, M, P, U, T, I, N, G. (Use the alphabetical order of the letters and insert them successively starting with the empty tree.)

   b. Assuming that the probabilities of searching for each of the keys (i.e., the letters) are the same, find the largest number and the average number of key comparisons for successful searches in this tree.

Author's hint:

7. a. Trace the algorithm for the input given (see Figure 6.8) for an example.

   b. Keep in mind that the number of key comparisons made in searching for a key in a 2-3 tree depends not only on its node's depth but also whether the key is the first or second one in the node.

#5 (25, not from textbook)

(sum of heights of nodes in a full tree) In this problem, we consider completely full binary trees with N nodes and height H (so that $N = 2^{H+1} - 1$)

(a) (5 points) Show that the sum of the heights of all of the nodes of such a tree can be expressed as $\sum_{k=0}^{H} k\,2^{H-k}$.

(b) (15 points) Prove by induction on H that the above sum of the heights of the nodes is N - H - 1. You may base your proof on the summation from part (a) (so you don't need to refer to trees at all), **or** you may do a "standard" binary tree induction based on the heights of the trees, using the definition that a non-empty binary tree has a root plus left and right subtrees. I find the tree approach more straightforward, but you may use the summation if you prefer.

(c) (3 points) What is the sum of the *depths* of all of the nodes in such a tree?

(d) (2 points) How does this apply to Heapsort analysis?

6.4 (6 - 10)

11. *Spaghetti sort* Imagine a handful of uncooked spaghetti, individual rods whose lengths represent numbers that need to be sorted.

    a. Outline a "spaghetti sort"—a sorting algorithm that takes advantage of this unorthodox representation.

    b. What does this example of computer science folklore (see [Dew93]) have to do with the topic of this chapter in general and heapsort in particular?

Author's hint:

11. Pick the spaghetti rods up in a bundle and place them end-down (i.e., vertically) onto a tabletop.

6.5 (7 - 3)

9. Is it a good idea to use a general-purpose polynomial evaluation algorithm such as Horner's rule to evaluate the polynomial $p(x) = x^n + x^{n-1} + \ldots + x + 1$?

Author's hint:

9. Use a formula for the sum of the terms of this special kind of a polynomial.