473 Levitin problems and hints   HW 09B

Problem 1: (2 points) 4.4.13 [5.5.7] Find J(40)–the solution to the Josephus problem for n = 40

Author's Hints: The fastest way to the answer the question is to use the formula that exploits the binary representation of n, which is mentioned at the end of the section.

Problem 2: (20 points) 4.4.15ab [5.5.9ab] For the Josephus problem,

a. (5) compute J(n) for n = 1, 2, …, 15.   **(Instructor note:** We started this table in class)

b. (15) discern a pattern in the solutions for the first fifteen values of n and prove its general validity by induction, based on the Josephus recurrence relations from the textbook.

Author's hints:

15. a. Use forward substitutions (see Appendix B) into the recurrence equations given in the text.

b. On observing the pattern in the first 15 values of n obtained in part (a), express it analytically. Then prove its validity by mathematical induction.

4.5.11a: This problem may be harder than first appears to be. You should provide an analysis in terms of m, n, and the (i, j) position of the moldy square For some values of (m, n, i, j), the first player can always win; for others the second player can always win. What is the winning strategy?

However, if you can't solve the general case, you may get some partial credit by solving the cases that you can solve, and writing about what you tried for other cases.

"Transform and conquer" is a good way to find a complete solution, so you may want to look ahead to Chapter 6 to give you some ideas of how "T & C" works.

In the past, several students said that this problem took them longer than any previous problem in the course.

Problem 3: (20 points) 4.5.11a[5.6.10a]

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?

Author's Hints:

10. 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.

Problem 4 (10 points) 6.1.5 [6.1.7]

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.

Problem 5: (10 points) 6.28c

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. You should compute and compare actual number of multiplications, not just say that both are Θ(n^3). Use division when you compare.

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.

Problem 6 (6 points) 6.3.7

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: (see instructor notes on next page)

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.

Instructor notes

You must (a) show the steps in constructing the tree, and (b) show the details of the average-case calculation.

Problem 7 (3 points) 6.3.8

Let T_B and $T_{2\text{-}3}$ be, respectively, a classical binary search tree and a 2-3 tree constructed for the same list of keys inserted in the corresponding trees in the same order. True or false: Searching for the same key in $T_{2\text{-}3}$ always takes fewer or the same number of key comparisons as searching in T_B?

Author's hint:

False; find a simple counterexample.

Problem 8 (3 points) 6.3.9

For a 2-3 tree containing real numbers, design an algorithm for computing the range (i.e., the difference between the largest and smallest numbers in the tree) and determine its worst-case efficiency.

Author's hint:

Where will the smallest and largest keys be located?