**473 Levitin problems and hints HW 09**

**Problems 1 and 2 are not from the textbook.**

**Problem 3:** 4.4.8 [5.5.2]

2. Consider ***ternary search***—the following algorithm for searching in a sorted array $A[0..n-1]$: if $n = 1$, simply compare the search key $K$ with the single element of the array; otherwise, search recursively by comparing $K$ with $A[\lfloor n/3 \rfloor]$, and if $K$ is larger, compare it with $A[\lfloor 2n/3 \rfloor]$ to determine in which third of the array to continue the search.

   a. What design technique is this algorithm based on?

   b. Set up a recurrence relation for the number of key comparisons in the worst case. (You may assume that $n = 3^k$.)

   c. Solve the recurrence for $n = 3^k$.

   d. Compare this algorithm's efficiency with that of binary search.

Author's Hints:

2. The algorithm is quite similar to binary search, of course. In the worst case, how many key comparisons does it make on each iteration and what fraction of the array remains to be processed?

**Problem 4:** 4.4.10[5.5.3]

3. a. Write a pseudocode for the divide-into-three algorithm for the fake-coin problem. (Make sure that your algorithm handles properly all values of $n$, not only those that are multiples of 3.)

   b. Set up a recurrence relation for the number of weighings in the divide-into-three algorithm for the fake-coin problem and solve it for $n = 3^k$.

   c. For large values of $n$, about how many times faster is this algorithm than the one based on dividing coins into two piles? (Your answer should not depend on $n$.)

Author's Hints:

3. While it is obvious how one needs to proceed if $n \bmod 3 = 0$ or $n \bmod 3 = 1$, it is somewhat less so if $n \bmod 3 = 2$.

**Problem 5:** 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 6:** 4.4.15 [5.5.9] For the Josephus problem,

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

b. discern a pattern in the solutions for the first fifteen values of n and prove its general validity.

**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 7:** 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.