Divide and conquer exercises

Problem 1 (Intermediate value)

You are given an array of integers $A[1...n]$, such that, for all i, $1 \leq i < n$, we have $|A[i]-A[i+1]| \leq 1$. Let $A[1] = x$ and $A[n] = y$, such that $x < y$. Design an algorithm to find an index j such that $A[j] = z$ for a given value of z, $x \leq z \leq y$.

Problem 2 (Majority element)

You are given as an input a list of n objects $A = [a_1, a_2, ..., a_n]$ where n is a power of 2. In $O(1)$ time you can check if two objects are the same or not. We say there is a majority element if there is an element which appears in the list A more than $n/2$ times. Give a **divide and conquer algorithm** to check if there is a majority element. Your algorithm should run in $O(n \log n)$ time.
Note, for two objects a_i and a_j you can check in constant time if $a_i = a_j$? But you cannot order them such as $a_i < a_j$ or $a_i > a_j$, that does not make sense with these objects. **Hence, you cannot sort these objects or find their median.** Explain your algorithm in words, and analyze the running time of your algorithm including stating the recurrence. And explain why your algorithm is correct.
Note, there is an $O(n)$ time algorithm for this problem, but you will not receive extra credit for that solution, so we suggest aiming for the simpler $O(n \log n)$ time algorithm.

Dynamic programming exercises

Problem 3 (LIS variation)

We call a sequence of integers $a_1, \ldots, a_n$ *noisy* when the signs of the differences between two consecutive terms in the sequence strictly alternate between + and - (the difference is never zero). So the sequence either follows $a_1 < a_2 > a_3 < a_4 > \ldots$ or it follows $a_1 > a_2 < a_3 > a_4 < \ldots$. An example of such a sequence is $2, 4, -1, 9, 0, 5, -2$. On the other hand, $2, 4, 7, 9, 0, 5, 5$ is not a *noisy* subsequence because the differences between the three consecutive elements $2, 4, 7$ do not alternate. Two 5's also show up at the end of the sequence causing the consecutive difference to be zero. You are given an array of integers $A = [a_1, \ldots, a_n]$. Find the length of the longest *noisy* subsequence in A.

Problem 4 [DPV 6.6]

Let us define a multiplication operation on three symbols a, b, c according to the following table; thus $ab = b$, $ba = c$, and so on. Notice that the multiplication operation defined by the table is neither associative nor commutative.

	a	b	c
a	b	b	a
b	c	b	a
c	a	c	c

Find an efficient algorithm that examines a string of these symbols, say *bbbbac*, and decides whether or not it is possible to parenthesize the string in such a way that the value of the resulting expression is a. For example, on input *bbbbac* your algorithm should return yes because $((b(bb))(ba))c = a$.

Problem 5 (Max-Weight Independent Set in Trees)

Assume that your graph is a tree $T = (V, E)$. Each vertex $u \in V$ also has a positive weight w_u. The max-weight independent set problem is to find an independent set S in the graph so that the total weight $\sum_{u \in V} w_u$ is as large as possible. Provide an efficient algorithm using dynamic programming to solve this problem on a tree.

(a) Is it true that every leaf belongs to some max-weight independent set? Explain your answer.
(b) Define the typical subproblem in words. State the recurrence for the subproblem in terms of smaller subproblems.
(c) Write pseudocode for your algorithm to solve this problem.
(d) Briefly explain/analyze the running time of your algorithm.