There are 4 problems, accounting for 100% in total.

Problem 1 (20%)**.** Assuming that an independent uniform hash function exists, design a randomized algorithm that solves the element uniqueness problem in expected $O(n)$ time. Justify your answer.

Problem 2 (30%)**.** Let $m \in \mathbb{N}$ be the number of slots in the hash table, $p \in \mathbb{N}$ be a prime number. Consider the family of hash functions

$$H_{p,m} \;:=\; \left\{\; h_{a,b} \colon a \in \mathbb{Z}_p^*,\; b \in \mathbb{Z}_p \;\right\},$$

where for any non-negative integer k,

$$h_{a,b}(k) \;:=\; (\;(ak+b) \bmod p\;) \;\text{ mod m}\;.$$

Prove that $H_{p,m}$ is universal and 2-independent, but is not 3-independent.

Problem 3 (20%)**.** Consider the following optimization problem.
There are N squares, numbered $1, 2, \ldots, N$. For each i with $1 \le i \le N$, the height of the square i is h_i. There is a frog who is initially at square 1. He will repeat the following action some number of times to reach the square N.

- If the frog is currently at square i, he can jump to $i+1$ or $i+2$ (but not backwards). For this move, a cost of $|h_j - h_i|$ is incurred, where j is the the square he jumps to.

Please use dynamic programming to find out the minimum cost the frog will incur to reach square N.

Problem 4 (30%)**.** Consider the following problem.

You are given an $a \times b$ rectangle, where a and b are integers, and your task is to cut it into squares. On each of the moves, you can pick a rectangle and cut it into two rectangles in a way such that the lengths of all sides remain integers.

Design an algorithm that computes the minimum number of moves to achieve this. Note that, your algorithm should run in time polynomial in a and b.

For example, if $a = 3$ and $b = 5$, then the minimum number of moves is 3. First you cut the rectangle into two rectangles, one with 3×3 and the other with 3×2. Next you cut the 3×2 rectangle into 2×2 and 1×2. Finally you cut the 1×2 rectangle into two 1×1 squares.