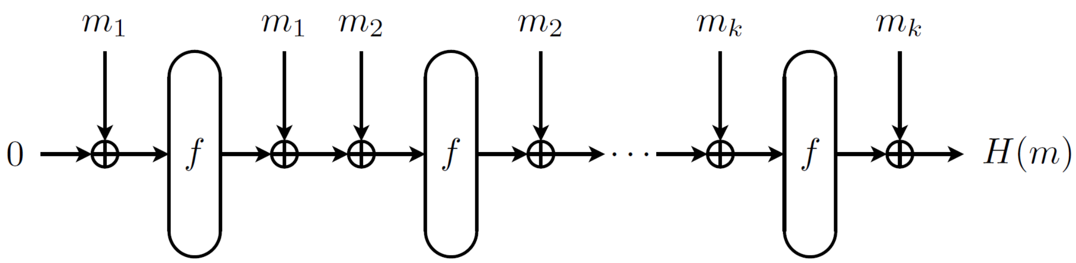
Problem 8. «Collisions»

Consider a hash function H that takes as its input a message m consisting of $k \cdot n$ bits and returns an n-bit hash value $H(m)$. The message m is at least one block long ($k \geqslant 1$), and can be split into k blocks of n bits each: m_1, m_2, …, m_k. Let f be a function which takes an n-bit input and returns an n-bit output. We will use $\oplus$ to denote the bitwise exclusive-or operator.

The hash function H is defined iteratively as follows:

$$h_i := m_i \oplus f(h_{i-1} \oplus m_i),$$

where all n bits of h_0 are zero, and $H(m) := h_k$. Below is an illustration of the hash function H.



A *collision* for H is defined as a pair of distinct messages (m, m') so that $H(m) = H(m')$. Given a message m and its corresponding hash value $H(m)$, a *second preimage* for H is defined as a message $m' \neq m$ so that $H(m) = H(m')$.

Suppose that f is a secret random function and that you have obtained $10 \cdot n$ random different pairs $(x, f(x))$ of argument and value of the function f. Under these restrictions, solve the following problems. Algorithms in Q1 and Q2 must give a solution with a high probability ($> 1/2$).

Q1 Propose an algorithm which finds a collision for H.

Q2 Propose an algorithm which, given a message m and its corresponding hash value $H(m)$, finds a second preimage m' for H.

Q3 Suppose that $n = 256$ bits and the message m is "`A random matrix is likely decent`". Find a second preimage m' for this message.

Remark 1. The text message is converted into a bit sequence as follows: first, each character is converted into a 8-bit integer according to the UTF-8 encoding, and then these integers are concatenated together using the big-endian ordering. For example, the string "`Hello`" is converted into the sequence of integers $(72, 101, 108, 108, 111)$ which then gives the following binary string: `0100100001100101011011000110110001101111`. You can give your answer to this task in the form of a binary sequence or a hexadecimal sequence.

Remark 2. You can evaluate the hash function H on any input message here. The message being hashed should be presented as either a binary sequence or a hexadecimal sequence, starting with a symbol `b` or `h` which specifies the representation. Here you can find a list of values of the function f on 512 different inputs (binary sequences are presented as integers).