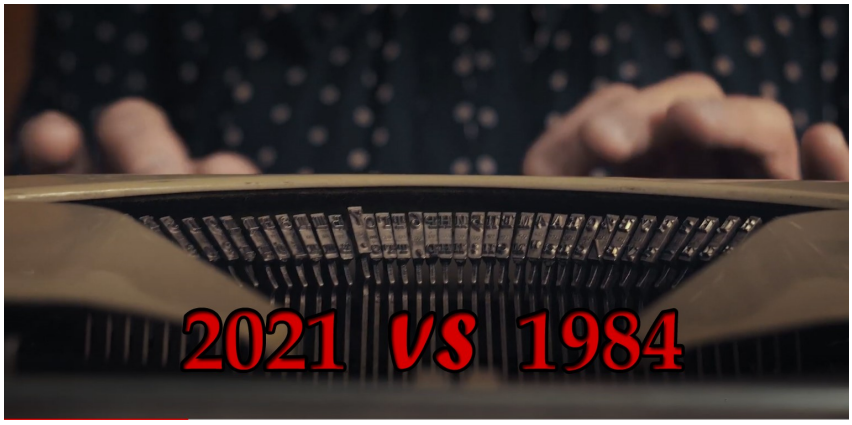
Problem 1. «2020»

A cipher machine `WINSTON` can transform a binary sequence in the following way. A sequence S is given, a cipher machine can add to S or remove from S any subsequence of the form 11, 101, 1001, 10...01. Also, it can add to S or remove from S any number of zeros.

When special agent Smith entered the room there were two identical `WINSTON` machines. He was curious to encrypt number 2020 and he tried to encrypt the number in it's binary form. The first cipher machine returned the binary form of number 1984, the second one returned the binary form of number 2021. Smith understood that one of the machines is broken. How did he know that?



Problem 2. «A secret house»

Here you can see a secret house.

Looking on it, could you understand what should be shown inside the frame left blank?

Problem 3. «Miller — Rabin revisited»

Bob decided to improve the famous Miller — Rabin primality test. The odd number n being tested is represented in the form $n-1 = 2^k 3^\ell m$, where m is not divisible by 2 or 3.

The modified primality test is the following:

1. Take a random $a \in \{2, \ldots, n-2\}$.
2. Put $a \leftarrow a^m \bmod n$. If $a = 1$, return "`PROBABLY PRIME`".
3. For $i = 0, 1, \ldots, \ell - 1$ do the following steps:
   (a) $b \leftarrow a^2 \bmod n$;
   (b) if $a + b + 1$ is divisible by n, return "`PROBABLY PRIME`";
   (c) $a \leftarrow ab \bmod n$.
4. For $i = 0, 1, \ldots, k - 1$ repeat:
   (a) if $a + 1$ is divisible by n, return "`PROBABLY PRIME`";
   (b) $a \leftarrow a^2 \bmod n$.
5. Return "`COMPOSITE`".

Q1 Prove that this algorithm does not fail, that is, not return "`COMPOSITE`", for a prime n.

Q2 **Bonus problem (extra scores, a special prize!)**

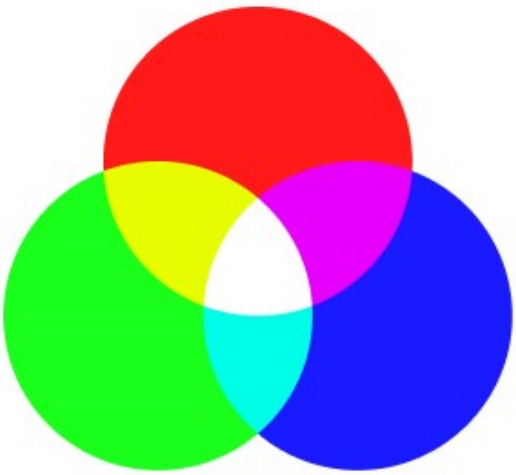
A composite integer n may be classified as "`PROBABLY PRIME`" by a mistake. It is known that for the usual Miller — Rabin test the error probability is less than $1/4$. Can this estimation be improved when we are switching to the described algorithm?

Remark. The expression $a \leftarrow a^m \bmod n$ means that a takes a new value that is equal to the remainder of dividing a^m by n.

Problem 4. «RGB»

Victor is studying the Moctod search server. Inside its software, he found two integer variables a and b that change their values when special search queries "`RED`", "`GREEN`" and "`BLUE`" are processed. More precisely, the pair (a, b) is changed to $(a+18b, 18a-b)$ when processing the query "`RED`", to $(17a+6b, -6a+17b)$ when processing "`GREEN`", and to $(-10a-15b, 15a-10b)$ when processing "`BLUE`". When any of a or b reaches a multiple of 324, it resets to 0. Whenever $(a, b) = (0, 0)$, the server crashes.

On server startup, the variables (a, b) are set to $(20, 20)$. Prove that the server will never crash with these initial values, regardless of the search queries processed.



Problem 5. «Mysterious event»

Mr. Bob is the editor in-chief of a well known magazine. He has many interests and activities in addition to work: meetings with bright people of politics and art, dancing, fishing, and even stenography and linguistics.

Every week, the magazine publishes a hard Sudoku on the last page. Mr. Bob likes this game too! So, it is a pleasure for him to personally analyze all solutions from the readers. He sits down in his office with a cup of coffee and looks through all the PNG-files with photos of solutions.

But suddenly Mr. Bob disappeared. The last solution he could see on his monitor was the following (here is a **link** to it, if you are interested in).

But what happened? Where is Mr. Bob?

Problem 6. «CPA game»

Suppose we have a system for the encryption of binary messages. The system has the following characteristics:

- Every message is divided into blocks of length n that are called plaintexts (it is supposed that the length of messages is divisible by n).
- The system employs a block cipher with the encryption function E in cipher block chaining (CBC) mode (see the picture below). A block, an initialization vector IV and a key lengths are equal to n. The result of encryption of the message is a concatenation of IV and the ciphertexts of all plaintexts it consists of.
- The IV for the first message is chosen randomly by using a secure pseudorandom number generator. The last ciphertext block of the i-th message is used as the IV for the $(i+1)$-st message.

Let Alice be an honest user of the system. Victor, an adversary, convinced her to play **chosen–plaintext attack game** (CPA game) with him.

The game is the following:

1. Alice selects a key $k \in \{0,1\}^n$ and chooses a bit $b \in \{0,1\}$.

2. Victor submits a sequence of q queries to Alice. For $i = 1, 2, \ldots, q$ repeat

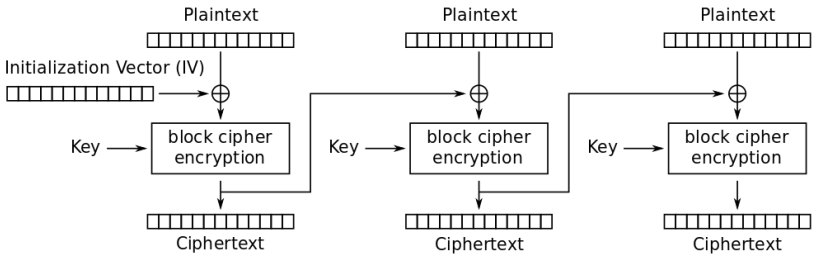
(a) Victor chooses a pair of messages, $m_{i,0}, m_{i,1}$ of the same length.

(b) Alice encrypts $m_{i,b}$ with the key k and gets c_i (the sequence of corresponding IV and ciphertexts). She sends c_i to Victor.

3. Victor outputs a bit $b^* \in \{0,1\}$.

Let W be the event that Victor guesses the bit, that is $b^* = b$. We define Victors's advantage with respect to E as CPAadv $:= |\Pr[\mathrm{W}] - 1/2|$. Victor wins the game if he can build an efficient algorithm such that CPAadv is not negligible.

Task. Construct an efficient probabilistic polynomial-time (PPT) algorithm that wins the CPA game against this implementation with an advantage close to $1/2$.



Cipher Block Chaining (CBC) mode encryption

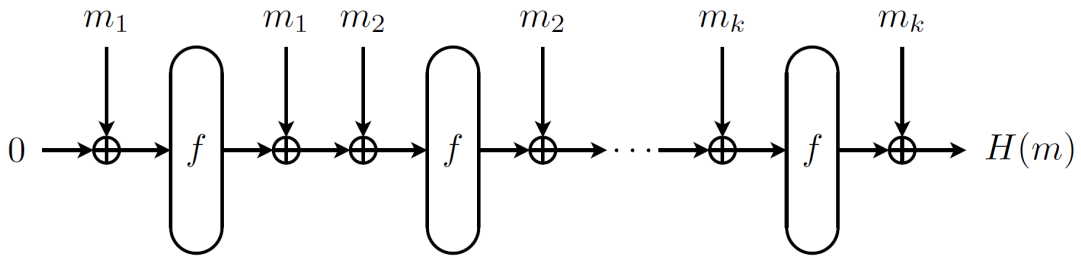
Problem 7. «Collisions»

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 a pair of distinct messages (m, 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, propose an algorithm which finds a collision for H with a high probability ($> 1/2$).