Problem 1. «The image set»

Special Prize from the Program Committee!

Let $\mathbb{F}_2$ be the finite field with two elements and n be any positive integer. Let $g(X)$ be an irreducible polynomial of degree n over $\mathbb{F}_2$. It is widely known that the set of equivalence classes of polynomials over $\mathbb{F}_2$ modulo $g(X)$ is a finite field of order 2^n; we denote it by $\mathbb{F}_{2^n}$.

Characterize in a non-straightforward way the image set (depending on n) of the function F over $\mathbb{F}_2^n$ defined as follows:

$$F(x) = x^3 + x.$$

That is, characterize in a way which brings additional information, for instance on its algebraic structure.

An example. For $n = 3$ we can take $g(X) = X^3 + X + 1$, then each element of the field $\mathbb{F}_{2^3}$ can be written as a polynomial of degree at most 2: $a_0 + a_1X + a_2X^2$, with $a_0, a_1, a_2 \in \mathbb{F}_2$. We can calculate the table of multiplication in $\mathbb{F}_{2^3}$ modulo $g(X)$, while the table of addition just corresponds to adding polynomials over $\mathbb{F}_2$. For example,

$$(1 + X + X^2) + (X + X^2) = 1,$$

$$(X + X^2)(1 + X^2) = X + X^2 + X^3 + X^4 = 1 + X \pmod{g(X)}.$$

Now we can calculate all the elements of the image set of $F(x)$. Indeed,

$$\{F(x) \mid x \in \mathbb{F}_{2^3}\} = \{0,\ 1,\ 1 + X,\ 1 + X^2,\ 1 + X + X^2\}.$$

Then we note that it is the union of $\{0\}$ and of the affine plane $1 + \{0, X, X^2, X + X^2\}$. In our case it is a desirable algebraic structure of this set.

You need to study this problem for an arbitrary n (or some partial cases).

Remark. Functions over the finite field of order 2^n are of great interest for using in cryptographic applications, for example, as S-boxes. For instance, AES S-box is based on the inverse function over $\mathbb{F}_{2^8}$. But in fact, there are many open problems in fields of finding new constructions and descriptions of cryptographically significant functions!

Problem 2. «TwinPeaks»

On Bob's smartphone there is a program that encrypts messages with the algorithm `TwinPeaks`. It works as follows:

1. It takes an input message P that is a hexadecimal string of length 32 and represents it as a binary word X of length 128.
2. Then X is divided into four 32-bits words a, b, c, d.
3. Then 6 iterations of the following transformation are applied:

   $$(a, b, c, d) \leftarrow \big(a + c + S(c + d), a + b + d + S(c + d), a + c + d, b + d + S(c + d)\big),$$

   where S is a secret permutation from $\mathbb{F}_2^{32}$ to itself and $+$ denotes the coordinate-wise sum modulo 2.
4. The word Y is obtained as a concatenation of a, b, c, d.
5. Finally, Y is converted to the hexadecimal string C of length 32. The algorithm gives C as the ciphertext for P.

Agent Cooper caught the ciphertext $C =$ `59A0D027D032B394A0A47A9ED19C98A8` send from Bob to Alice and decided to decrypt it.

In order to solve this problem agent Cooper captured also Bob's smartphone with `TwinPeaks` realization! Here it is. Now Cooper (and you too) can encrypt any messages with `TwinPeaks` but still can not decrypt any one.

Help Cooper to decrypt C.

Problem 3. «An addition chain»

In many cryptographic systems we need to calculate the value $B = A^c \bmod p$, where A is an integer, $1 \leqslant A \leqslant p-1$, c is an arbitrary positive integer, and p is a large prime number. One possible way of reducing the computational load of calculating is to minimize the total number of multiplications required to compute the exponentiation. Since the exponent in equation is additive, the problem of computing powers of the base element A can also be formulated as an addition calculation, for which addition chains are used.

An **addition chain** for an integer n is a sequence of positive integers

$$a_0 = 1, a_1, \ldots, a_{r-1}, a_r = n,$$

where r is a positive integer (that is called the **length** of the addition chain) and the following relation holds for all i, $1 \leqslant i \leqslant r$:

$$a_i = a_j + a_k \text{ for some } k, j \text{ such that } k \leqslant j < i.$$

Find an addition chain of length as small as possible for the value

$$2^{127} - 3.$$

The solution should be submitted as a list of values occurring in the chain and a description how you found the solution.

An example. For the value 15 the shortest additional chain has length 5 and its list of values is $1, 2, 3, 6, 12, 15$. So, to optimally calculate $B = A^{15} \bmod p$, one can use just five multiplications:

$$\begin{aligned} A^2 &= A \cdot A \mod p \\ A^3 &= A^2 \cdot A \mod p, \\ A^6 &= A^3 \cdot A^3 \mod p, \\ A^{12} &= A^6 \cdot A^6 \mod p, \\ A^{15} &= A^{12} \cdot A^3 \mod p. \end{aligned}$$

Problem 4. «Hash function FNV2»

The FNV2 hash function is derived from the function FNV-1a. FNV2 processes a message x composed of bytes $x_1, x_2, \ldots, x_n \in \{0, 1, \ldots, 255\}$ in the following way:

1) $h \leftarrow h_0$;

2) for $i = 1, 2, \ldots, n$: $h \leftarrow (h + x_i)g \bmod 2^{128}$;

3) return h.

Here $h_0 = 144066263297769815596495629667062367629$, $g = 2^{88} + 315$.

Find a collision, that is, two different messages x and x' such that $\texttt{FNV2}(x) = \texttt{FNV2}(x')$. Collisions on short messages and collisions that are obtained without intensive calculations are welcomed. Supply your answer as a pair of two hexadecimal strings which encode bytes of colliding messages.

Problem 5. «A music lover»

As usual Alex listens to music on the way to university. He chooses it applying one secret code to the second one in his mind. Could you understand what music he is listening to right now?

Remarks.

1. You should invent a way how to apply one code to another.
2. Some arithmetic operations can also be used.

Problem 6. «Boolean hidden shift and quantum computings»

Special Prize from the Program Committee!

The following long-standing problem is known. Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be a given Boolean function. Determine the hidden nonzero shift $a \in \mathbb{F}_2^n$ for the function, i.e. a vector such that $f_a(x) = f(x \oplus a)$ for all $x \in \mathbb{F}_2^n$. And do it having a limited access to an oracle. Such a problem is called the **Boolean hidden shift problem** (BHSP).

In order to solve this problem on a quantum computer, an oracle that computes the shifted function in the phase is used. This oracle can be implemented using only one query to an oracle that computes the function in a register. The phase oracle is $O_{f_a} : |x\rangle \mapsto (-1)^{f(x \oplus a)}|x\rangle$. The **quantum query complexity** is the minimum number of oracle O_{f_a} accesses needed in the worst case to solve the problem.

There are two classes of Boolean functions for which the quantum query complexity is minimal and maximal respectively:

- for any bent function, i. e. a function in even number of variables that is on the maximal possible Hamming distance from the set of all affine functions, one quantum query suffices to solve the problem exactly [1];
- for any delta function, i. e. $f(x) = \delta_{x,x_0}$ for some $x_0 \in \mathbb{F}_2^n$, the quantum query complexity is $\Theta\left(2^{n/2}\right)$, which is equivalent to Grover's search [2,3].

For any Boolean function f in n variables

$$Q\left(BHSP_f\right) = O\left(2^{n/2}\right),$$

where $Q\left(BHSP_f\right)$ is the bounded error quantum query complexity of the BHSP for f. Moreover, it holds [4]

$$Q\left(BHSP_f\right) \leqslant \frac{\pi}{4}\frac{2^{n/2}}{\sqrt{\mathrm{wt}(f)}} + O\left(\sqrt{\mathrm{wt}(f)}\right),$$

when $1 \leqslant \mathrm{wt}(f) \leqslant 2^{n-1}$, where $\mathrm{wt}(f)$ is the Hamming weight of f.

Turn to the next page.

The problem to solve is the following: identify natural classes of Boolean functions in even number of variables lying between the two extreme cases of bent and delta functions and characterize the quantum query complexity of the BHSP for these functions [4].

[1] M. Rotteler. Quantum algorithms for highly non-linear Boolean functions // Proceedings of the 21st ACM-SIAM Symposium on Discrete Algorithms (SODA 2010), pp. 448–457. SIAM, 2010.

[2] L. K. Grover. A fast quantum mechanical algorithm for database search // Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC 1996), pp. 212–219. ACM, 1996.

[3] C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani. Strengths and weaknesses of quantum computing // SIAM Journal on Computing, V. 26, I. 5, pp. 1510–1523, 1997.

[4] A. M. Childs, R. Kothari, M. Ozols, M. Roetteler. Easy and Hard Functions for the Boolean Hidden Shift Problem // Proceedings of TQC 2013, LIPIcs, V. 22, pp. 50–79, 2013.

Problem 7. «One more parameter»

There are several parameters in cryptanalysis of block ciphers that are used to measure the diffusion strength. In this problem, we study properties of one of them.

Let n, m be positive integers. Let $a = (a_1, \ldots, a_m)$ be a vector with coordinates a_i taken from the finite field $\mathbb{F}_2$. Denote the number of nonzero coordinates a_i, $i = 1, \ldots, m$, by $\mathrm{wt}(a)$ and call this number the **weight** of the vector a.

The inner product of $a = (a_1, \ldots, a_m)$ and $b = (b_1, \ldots, b_m)$ in $\mathbb{F}_2^m$ is defined as

$$a \cdot b = a_1 b_1 \oplus \ldots \oplus a_m b_m.$$

For a Boolean function $f : \mathbb{F}_2^m \to \mathbb{F}_2$, we define the function *weight* wt as follows:

$$\mathrm{wt}(f) = \#\{a \in \mathbb{F}_2^m \mid f(a) = 1\}$$

The **special parameter** Q of a vectorial Boolean function $\varphi : \mathbb{F}_2^m \to \mathbb{F}_2^m$ is defined to be

$$Q(\varphi) = \min_{a,\, b,\, b \neq 0,\, \mathrm{wt}(a \cdot x \oplus b \cdot \varphi(x)) \neq 2^{m-1}} \{\mathrm{wt}(a) + \mathrm{wt}(b)\}.$$

- Rewrite (simplify) the definition of $Q(\varphi)$ when the function φ is linear (recall that a function ℓ is linear if $\ell(x \oplus y) = \ell(x) \oplus \ell(y)$ for any x, y).
- Rewrite the definition of $Q(\varphi)$ in terms of linear codes, when the linear function φ is given by an $m \times m$ matrix M over $\mathbb{F}_2$, i. e. $\varphi(x) = Mx$.
- Find the tight upper bound for $Q(\varphi)$ as a function of m.
- Can you give an example of the function φ with the maximal possible value of Q?

Problem 8. «Scientists»

Two young cryptographers and very curious students Alice and Bob studied different cryptosystems and attacks on them. At the same time they were very interested in biographies of famous scientists and found out one interesting **property** that can be used in cryptosystems. They choose three pair of scientists:

Charles Darwin and **Michael Faraday**,
Werner Heisenberg and **Johannes Kepler**,
Hans Christian Orsted and **Mikhail Lomonosov**.

Alice and Bob choose a cryptosystem and an attack they would like to study. They constructed three sets of parameters for the cryptosystem: one set according to each pair of scientists. Then Alice choose a phrase consisting of 18 English letters (spaces were omitted) and divided it into three parts of 6 letters. She represented each part as a hexadecimal number using ASCII code. Alice encrypted the first part by the cryptosystem for each set of parameters, then the same actions she made for the second and the third parts. Finally, Alice got the following three groups of three ciphertexts (in hexadecimal notation):

	Part 1	Part 2	Part 3
Set of parameters 1	`2512 1F5A 0079`	`B494 222D 3E1C`	`275E B751 4FDB`
Set of parameters 2	`3D0D 6812 0443`	`5111 5BFD 9398`	`0815 6223 2698`
Set of parameters 3	`1EDC 4856 8CE2`	`9C18 2A32 B9AB`	`9A1C AD5C 25D7`

and asked Bob to decrypt it using the attack! Bob successfully read the secret phrase. Could you

- find the property like Alice and Bob,
- understand what is the cryptosystem and the attack chosen,
- decrypt the ciphertext by applying this attack?

*What word should be added at the beginning of the decrypted text according to the famous words of Mikhail Lomonosov?

Problem 9. «Masking»

It is known that there are attacks on cryptosystems that use information obtained from the physical implementation of a cryptosystem, for example, timing information, power consumption, electromagnetic leaks or even sound. To protect cryptosystems from such attacks cryptographers can use a countermeasure known as **masking**.

Correlation immune Boolean functions can reduce the masking cost. Therefore, we need to search for Boolean functions satisfying the following conditions: they should have **small Hamming weight** and **high correlation immunity**.

Let f be a non-constant Boolean function in 12 variables of correlation immunity 6.

- What is the lowest possible Hamming weight k of f?
- Give an example of such a function f with Hamming weight k.

Remark I. Hamming weight $\mathrm{wt}(f)$ of a Boolean function f in n variables is the number of vectors $x \in \mathbb{F}_2^n$ such that $f(x) = 1$.

Remark II. A Boolean function f in n variables is called *correlation immune of order* t, where t is an integer such that $1 \leqslant t \leqslant n$, if

$$\mathrm{wt}(f_{i_1,\ldots,i_t}^{a_1,\ldots,a_t}) = \mathrm{wt}(f)/2^t$$

for any set of indexes $1 \leqslant i_1 < \ldots < i_t \leqslant n$ and any set of values $a_1, \ldots, a_t \in \mathbb{F}_2$. Here $f_{i_1,\ldots,i_t}^{a_1,\ldots,a_t}$ denote the subfunction of f in $n - t$ variables that is obtained from $f(x_1, \ldots, x_n)$ by fixing each variable x_{i_k} by the value a_k, $1 \leqslant k \leqslant t$.

Problem 10. «PIN code»

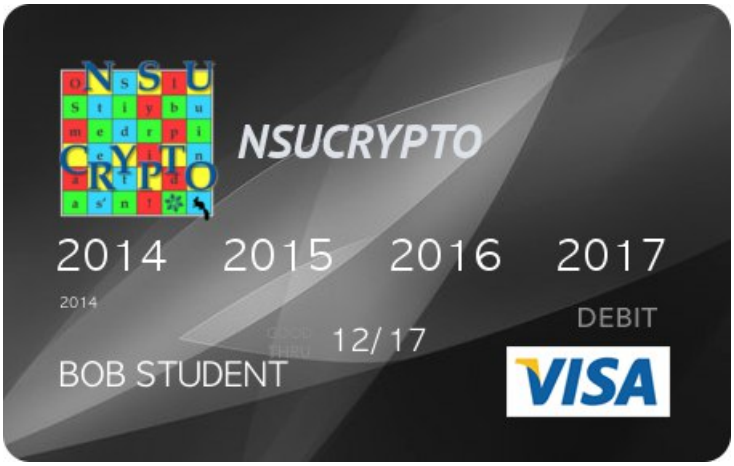
A PIN code $P = \overline{p_1 p_2 \ldots}$ is an arbitrary number consisting of a few pairwise different digits in ascending order ($p_1 < p_2 < \ldots$). Bob got his personal PIN code in the bank, but he decided that the code is not secure enough and changed it in the following way:

1. Bob multiplied his PIN code P by 999 and obtained the number $A = \overline{a_1 a_2 \ldots}$;

2. Then he found the sum of all digits of A: $a_1 + a_2 + \ldots = S = \overline{s_1 s_2 \ldots}$;

3. Finally, he took all digits (starting from 0) that are smaller than s_1, sorted them in ascending order and inserted between digits s_1 and s_2 in the number S. Resulting number P' is Bob's new PIN code. For example, if S was 345, then, after such insertion we obtain $P' = 301245$.

Find the new code P'!



Remarks.

1. The picture below the task is not linked with the task and was used only for decorative purposes. So, there is no any PIN codes or digits related to the problem.
2. A PIN code is an ARBITRARY number of ARBITRARY length (not 4).

Problem 11. «Useful Proof-of-work for blockchains»

Special Prize from the Program Committee!

Proof-of-work system is one of the key parts of modern blockchain-based platforms implementations, like cryptocurrency **Bitcoin** or **Ethereum**. Proof-of-work means that user is required to perform some work in order to request some service from the system, e. g. to send an e-mail or to create a new block of transactions for the blockchain.

For example, in Bitcoin system, if some user wants to create a block of transactions and add it to the chain, hash value of his block must satisfy certain condition, which can be achieved by iterating special variable X inside the block many times and checking the resulting hash value on every iteration.

What is important about the problem in a proof-of-work system, is that

- It is known that the solution for the problem exists, and it is also known how many iterations (on average) are required to find it, using best known algorithm $\mathcal{A}$;
- There are no algorithms for solving the problem, that perform significantly better than $\mathcal{A}$; it is believed also that such algorithms will not be found soon;
- Problem depends on some input data I, so you can not find solutions for the problem in advance (before input I is known) and then use these solutions without performing any work;
- Given a problem and a solution to it, it is easy to verify that provided solution is correct.

Unfortunately, solving the problem of finding specific hash values (used in Bitcoin and Ethereum) does not yield any information that is useful outside the system, therefore tremendous amounts of calculations performed to solve the problem are wasted.

Some other implementations of proof-of-work system solve this issue. For example, solutions of proof-of-work problem used in cryptocurrency **Primecoin** give us special chains of prime numbers, useful for scientific research.

Turn to the next page.

Your task is to construct a problem **P** that can be used in a proof-of-work system, such that information obtained in the proccess of solving it can be useful outside the system. More formally:

- **P** is, in fact, a family of problems, parametrized by two variables: I (input data, you can assume that I is a 256 bit string, or introduce other sensible formats), and C (complexity, e. g. some positive integer). For fixed input and complexity, $\mathbf{P}(I, C)$ is a problem that can be solved by using some algorithm $\mathcal{A}$ (should be provided in your solution to this task). It should not be possible to find a provable solution for the problem $\mathbf{P}(I, C)$ if I is not known;
- Average time T (amount of computational steps or iterations), required to find a solution of $\mathbf{P}(I, C)$ using algorithm $\mathcal{A}$ is known (assuming input data I is chosen randomly and uniformly), and depends on C, so $T = T(C)$, and $T(C)$ can be made very small, infeasibly large, or something in-between by adjusting complexity variable C;
- It should be easy to verify whether any provided solution is correct or not;
- Any kind of proof that there are likely no significantly better algorithms for solving **P** than the given algorithm $\mathcal{A}$, is desirable. For example, proof that proposed problem is $\mathcal{NP}$-hard, or any other considerations;
- You should describe how information obtained in the process of solving **P** can be useful outside of the proof-of-work system.

For example, in Bitcoin system, $\mathbf{P}(I, C)$ is a problem of finding an integer X such that if we apply SHA-256 hash function to the pair (I, X) twice, resulting hash value, represented as an integer, will be not greater than C. Here C is a nonnegative integer, defining complexity of the problem, and I — a block header, containing information about all transactions included in it, along with some other information — is an input.