Problem 1. «A 1024-bit key»

Alice has a 1024-bit key for a symmetric cipher (the key consists of 0s and 1s). Alice is afraid of malefactors, so she changes her key everyday in the following way:

1. Alice chooses a subsequence of key bits such that the first bit and the last bit are equal to 0. She also can choose a subsequence of length 1 that contains only 0.

2. Alice inverts all the bits in this subsequence (0 turns into 1 and vice versa); bits outside of this subsequence remain as they are.

Prove that the process will stop. Find the key that will be obtained by Alice in the end of the process.

Example of an operation. $11001\underbrace{01101110}011\ldots$ turns to $11001\underbrace{10010001}011\ldots$

Problem 2. «Sharing»

Bob is interested in studying mathematical countermeasures to side-channel attacks on block ciphers. He found out that techniques such as special sharings of functions can be applied against such attacks. Now he is thinking about the following mathematical problem in this approach.

Let $\mathcal{F}$ denote the set of **invertible functions** (**permutations**) from $\mathbb{F}_2^4$ to $\mathbb{F}_2^4$ and $\mathcal{F}^n$ denote the set of invertible functions from $(\mathbb{F}_2^4)^n$ to $(\mathbb{F}_2^4)^n$. Let $F \in \mathcal{F}^n$ be

$$F(x_1, x_2, \ldots, x_n) = (F_1(x_1, x_2, \ldots, x_n), F_2(x_1, x_2, \ldots, x_n) \ldots, F_n(x_1, x_2, \ldots, x_n)),$$

with component functions $F_i : (\mathbb{F}_2^4)^n \to \mathbb{F}_2^4$, $i = 1, \ldots, n$.

For any $f \in \mathcal{F}$, a function $F \in \mathcal{F}^n$ is called a **sharing** of f if

$$\sum_{i=1}^{n} F_i(x_1, x_2, \ldots, x_n) = f\left(\sum_{i=1}^{n} x_i\right) \quad \text{for all} \quad (x_1, x_2, \ldots, x_n) \in (\mathbb{F}_2^4)^n.$$

Moreover, F is a **non-complete** sharing of f if F is a sharing of f with the additional property that each component function F_i is independent of x_i.

Bob needs your help to study functions for which a non-complete sharing exists. Find answers to the following questions!

Q1 Let $\mathcal{A}$ denote the set of **affine functions** from $\mathbb{F}_2^4$ to $\mathbb{F}_2^4$. Two functions $f, g \in \mathcal{F}$ are **affine equivalent** if there exist $a, b \in \mathcal{A}$ such that $g = b \circ f \circ a$.

Let f, g be two functions in the same affine equivalence class of $\mathcal{F}$ and let F be a non-complete sharing of f. Derive from F a non-complete sharing for g.

All functions of the same affine equivalence class have the same degree. It is known [1] that this equivalence relation divides $\mathcal{F}$ into 302 classes: 1 class corresponds to $\mathcal{A}$, 6 classes contain quadratic functions, 295 classes contain cubic functions.

Also, Bob knows that when $n \geqslant 5$, there exists a non-complete sharing for each $f \in \mathcal{F}$ (it can be shown by construction). When $n = 2$ a non-complete sharing exists only for the functions in $\mathcal{A}$. When $n = 3$, non-complete sharings exist for $\mathcal{A}$ and also for 5 out of the 6 equivalence classes containing quadratic functions. When $n = 4$, non-complete sharings exist for $\mathcal{A}$, for all 6 quadratic equivalence classes and for 5 cubic classes.

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

Find a concise mathematical property that a function $f \in \mathcal{F}$ must have in order that a non-complete sharing F exists for $n = 3, 4$.

Q3 Bonus problem (extra scores, a special prize!)

Generalize to functions over $\mathbb{F}_2^5$, $\mathbb{F}_2^6$.

[1] C. De Canni‘ere. Analysis and Design of Symmetric Encrytption Algorithms, Ph.D. thesis, 2007.

Problem 3. «Factoring in 2019»

Nicole is learning about the RSA cryptosystem. She has chosen random 500-bit prime numbers p and q, $2^{499} \leqslant p, q < 2^{500}$, and computed $n = p \cdot q$. Being a curious and creative person, she has also combined the three numbers in funny ways. Her favorite one is an integer h such that

$$h \equiv 3^{2019} p^2 + 5^{2019} q^2 \pmod{n^2 + 8 \cdot 2019}.$$

Unfortunately, she has lost the paper where she wrote the two prime numbers. Luckily, she remembers n and h. Help Nicole to recover p and q.

$$\begin{aligned} n = {} & 40763613025504836845249840044831561583564626405535158138667037 \\ & 18791672670905308860844304055285019651507728831663677166092475 \\ & 16155419756121537288444995708421977847213953345126368990185271 \\ & 10259760189356588305406519080647582874212687596214191915933827 \\ & 672520947172224181322892513146475004919963234000002019, \end{aligned}$$

$$\begin{aligned} h = {} & 78307999278336577586961528110240026923828914927526911949501196 \\ & 64549497756373569985393554661132717198368717093111812566649031 \\ & 17342818449633588647098544612151278035131454234786653136500887 \\ & 08830470996542888912418213532073622903727205396807848603735835 \\ & 72653630883685906916701587362236649126895719656663293825501223 \\ & 97088799629252601249428062432254738935764304610281613264225641 \\ & 74990272864680012560095992125783832230234589257650929348364268 \\ & 48117494065463529201859600747521892957258104033195441014023432 \\ & 36581529201392185327635674923459290749241831590661903965132514 \\ & 2154451518308886658505820006667836934411881. \end{aligned}$$

Problem 4. «TwinPeaks3»

As Bob's previous cipher TwinPeaks2 (NSUCRYPTO-2018) was broken again, he finally decided to read some books on cryptography. His new cipher is now inspired by practical ciphers, while the number of rounds was reduced a bit for better performance.

Not only the best techniques were adopted by Bob, but also he decided to enhance his cipher by security through obscurity, so the round functions are now unknown. The only thing known about these functions is that they are the same for odd and even rounds.

New Bob's cipher works as follows. A message X is represented as a binary word of length 128. It is divided into four 32-bit words a, b, c, d and then the following round transformation is applied 32 times:

$$(a, b, c, d) \leftarrow (b, c, d, a \oplus (F_i(b, c, d)))$$
$$F_i = F_1 \text{ for odd rounds and } F_i = F_2 \text{ for the rest.}$$

Here F_1 and F_2 are secret functions accepting three 32-bit words and returning one word; and $\oplus$ is the binary bitwise XOR. The concatenation of the final a, b, c, d is the resulting ciphertext Y for the message X.

Agent Cooper again wants to read Bob's messages. He caught the ciphertext

$$Y = \texttt{e473f19a247429ab33b66268d57dd241}$$

(the ciphertext is given in hexadecimal notation, the first byte is e4).

He was also able to gain access to Bob's testing server with encryption and decryption routines, using the secret key. Here it is. Unfortunately, the version of software available on this server is not final. So, the decryption routine is incomplete and only uses keys in the reverse order, which is not sufficient for decryption:

$$(a, b, c, d) \leftarrow (b, c, d, a \oplus (F_i(b, c, d)))$$
$$F_i = F_2 \text{ for odd rounds and } F_i = F_1 \text{ for the rest.}$$

The server can also process multiple blocks of text at a time: they will be processed one-by-one and then concatenated, as in the regular ECB cipher mode of operation. Ciphertexts and plaintexts are given and processed by the server in hexadecimal notation.

Help Cooper to decrypt Y.

Problem 5. «Curl27»

Bob is developing the 3OTA infrastructure and has designed a new hash function Curl27 for it. A distinguishing feature of the infrastructure is the ternary logic: trits from the set $\mathsf{T} = \{0, 1, -1\}$ are used instead of bits, ternary strings and words are used instead of binary ones. The Curl27 hash function is defined below. Its implementation in Java can be found here.

Find a collision for Curl27, that is, different ternary strings X and X' such that $\mathsf{Curl27}(X) = \mathsf{Curl27}(X')$. Submit colliding strings as two lines of trits separated by commas. An example of a (wrong!) solution is:

```
-1,1,0,1,1,0
-1,-1,1,0,1,1,-1,0
```

Description of Curl27. The Curl27 function maps a ternary string X of arbitrary length to a hash value from T^{243}. When hashing, an auxiliary sponge function Curl27-f: $\mathsf{T}^{729} \to \mathsf{T}^{729}$ is used. The hashing algorithm:

1. Pad X with zeros to make its length a multiple of 243. Divide the resulting string into blocks $X_1, X_2, \ldots, X_d \in \mathsf{T}^{243}$.
2. Prepare the state $W = W_0 W_1 W_2 \in \mathsf{T}^{729}$ consisting of words $W_i \in \mathsf{T}^{243}$. Initialize the state by filling W_0 and W_2 with zeros and W_1 with the encoded initial (before padding) length of X. The length is encoded by a ternary word according to the little-endian conventions: less significant trits go first. For example, the length $25 = 1 - 3^1 + 3^3$ is presented by the word $\underbrace{1\bar{1}01000\ldots0}_{243}$.

   Here $\bar{1}$ stands for -1.
3. For $i = 1, 2, \ldots, d$, do: $W_0 \leftarrow X_i$, $W \leftarrow \mathsf{Curl27\text{-}f}(W)$.
4. Return W_0.

Description of Curl27-f. In Curl27-f the S-box

$$S\colon \mathsf{T}^3 \to \mathsf{T}^3, \quad (a, b, c) \mapsto (F(a, b, c), F(b, c, a), F(c, a, b))$$

is used. Here

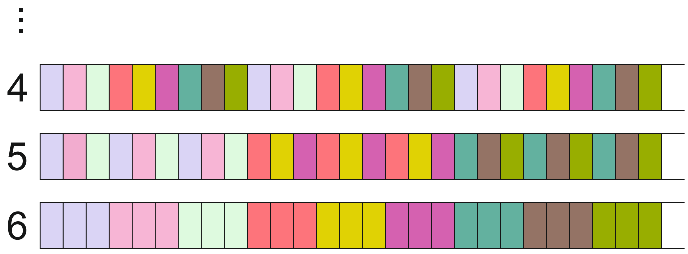
$$F(a, b, c) = a^2b^2c + a^2bc^2 - ab^2c^2 + a^2b^2 - a^2bc + a^2c^2 + ab^2c - a^2c + ab^2 - ac^2 + b^2c + bc^2 - a^2 - b^2 + bc - c^2 - c + 1,$$

where the calculations are carried out modulo 3 while the residue 2 is represented by the trit -1.

To transform the state W, 27 rounds are performed. A round consists of 6 steps. At each step triplets of trits of W are grouped in a certain way. Then each triplet (a, b, c) is replaced with $S(a, b, c)$.

Turn to the next page.

Groupings are organized as follows (see figure). At the first step, the state is divided into 3 words of 243 trits. Trits of these words in the same positions are grouped. In the second step, the state is divided into 9 words of 81 trits. Trits of the 1st, 2nd and 3rd words in the same positions are grouped, then trits of the 4th, 5th and 6th words, and so on. After that, the state is divided into words of length 27, then length 9, then length 3 while maintaining the logic of groupings. In the last sixth step, consecutive triplets of trits are grouped.



Groupings (3 last steps, grouped trits are painted the same color

Bonus problem (extra scores, a special prize!).

Find a collision when the state is initialized in a different way: now W_0, W_2 are not filled with zeros, the word $\underbrace{01\bar{1}01\bar{1}\ldots01\bar{1}}_{243}$ is written in each of them instead.

Problem 6. «8-bit S-box»

Permutations S of the set $\{0,1\}^n$ or $\mathbb{F}_2^n$ are usually called *n-bit S-boxes*. We will focus on the following cryptographic properties of S-boxes:

1. **The (minimal) algebraic degree** of S, denoted by $\deg(S)$, is the minimum of algebraic degrees of all component functions of S.
2. **The nonlinearity of** S, denoted by $\mathrm{nl}(S)$, is the minimal Hamming distance between all component functions of S and the set of all affine functions.
3. **The differential uniformity** of S, denoted by $\mathrm{du}(S)$ is the maximal number of solutions of the equation $S(x) \oplus S(x \oplus \alpha) = \beta$ for any nonzero vector α and any vector β.
4. **The (graph) algebraic immunity** of S, denoted by $\mathrm{ai}(S)$, is the minimal algebraic degree of all nonzero Boolean functions f in $2n$ variables such that $f(x,y) = 0$ for any $x \in \mathbb{F}_2^n$ and $y = S(x)$.

In modern symmetric cryptography, S-boxes of dimension $n = 8$ are probably the most popular. For example, such an S-box is used in the AES block cipher. The characteristics of S_{AES}:

$$(\deg, \mathrm{nl}, \mathrm{du}, \mathrm{ai})(S_{\text{AES}}) = (7, 112, 4, 2).$$

The value $\mathrm{ai}(S_{\text{AES}}) = 2$ means that S_{AES} (and the whole AES) can be compactly described by quadratic equations. This can be a weakness in the context of algebraic attacks.

Imposing the restrictions $(\deg, \mathrm{ai})(S) = (7, 3)$ (optimal values), we need to maximize $\mathrm{nl}(S)$ and minimize $\mathrm{du}(S)$. The current best result [1,2] is

$$(\deg, \mathrm{nl}, \mathrm{du}, \mathrm{ai})(S) = (7, 108, 6, 3).$$

Problem for a special prize! You need to improve this result:

find an 8-bit S with $\mathrm{nl}(S) > 108$ and/or $\mathrm{du}(S) < 6$ while preserving $\deg(S) = 7$ and $\mathrm{ai}(S) = 3$.

Remark. Let us recall the relevant definitions.

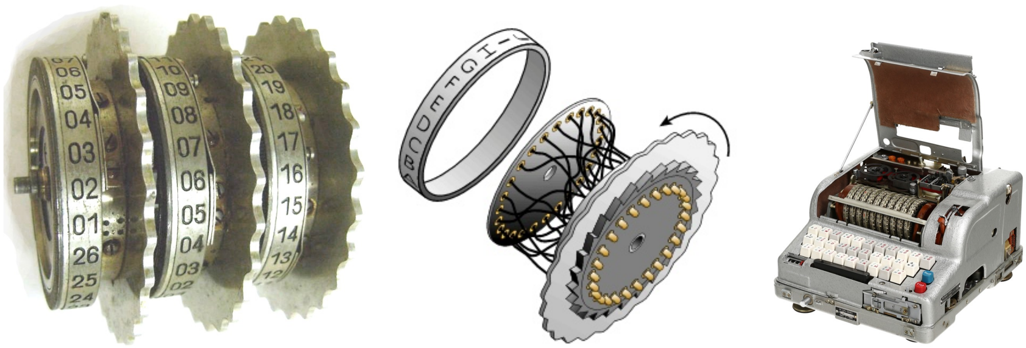
1. A Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ can be uniquely represented in the *algebraic normal form* (ANF) in the following way: $f(x) = \bigoplus_{I \in \mathcal{P}(N)} a_I \big(\prod_{i \in I} x_i\big)$, where $\mathcal{P}(N)$ is the power set of $N = \{1, \ldots, n\}$ and $a_I \in \mathbb{F}_2$.
2. The *algebraic degree* of f is degree of its ANF: $\deg(f) = \max\{|I| : \ a_I \neq 0, \ I \in \mathcal{P}(N)\}$.
3. Boolean functions of the algebraic degree not more than 1 are called *affine*.
4. The Hamming distance between Boolean functions f and g is the number of vectors $x \in \mathbb{F}_2^n$ such that $f(x) \neq g(x)$.
5. A function $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$ can be given as $S = (s_1, \ldots, s_n)$, where s_i is a Boolean function; a nontrivial linear combination of $s_1, \ldots, s_n$ is a *component* function of S.

[1] R.A. de la Cruz Jimènez. Generation of 8-Bit S-Boxes Having Almost Optimal Cryptographic Properties Using Smaller 4-Bit S-Boxes and Finite Field Multiplication. In: Lange T., Dunkelman O. (eds) Progress in Cryptology – LATINCRYPT 2017. LNCS, 2019, vol 11368, pp 191–206.

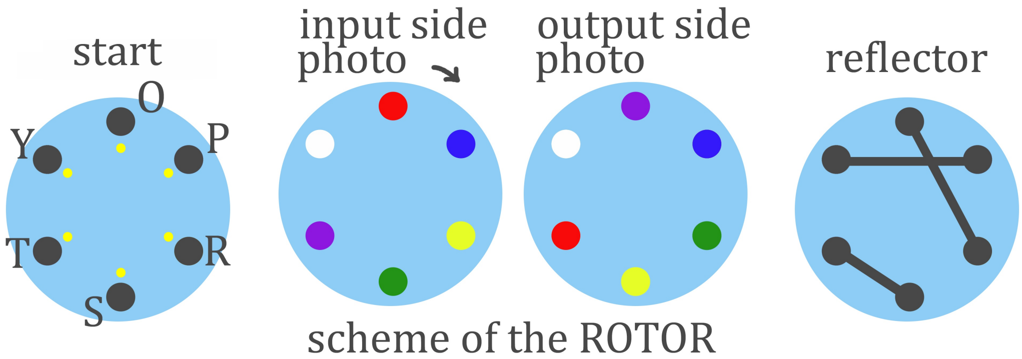
[2] D. B. Fomin. New classes of 8-bit permutations based on a butterfly structure. Math. vopr. kript. 2019, vol 10(2), pp 169–180. https://ctcrypt.ru/files/files/2018/09_Fomin.pdf.

Problem 7. «A rotor machine»

In one country rotor machines were very useful for encryption of information.



Eve knows that for some secret communication a simple rotor machine was used. It works with letters O, P, R, S, T, Y only and has an input circle with lamps (start), one rotor and a reflector. See the scheme below.



The input circle and the reflector are fixed in their positions while the rotor can be in one of 6 possible positions. After pressing a button on a keyboard, an electrical signal corresponding to the letter goes through the machine, comes back to the input circle, and the appropriate lamp shows the result of encryption. After each letter is encrypted, the rotor turns right (i. e. clockwise) on 60 degrees. Points of different colors on the rotor sides indicate different noncrossing signal lines within the rotor.

For instance, if the rotor is fixed as shown on the picture above, then if you press the button O, it will be encrypted as T (the signal enters the rotor via red point, is reflected and then comes back via purple line). If you press O again, it will be encrypted as R. If you press T then, you will get S and so on.

Eve intercepted a secret message: TRRYSSPRYRYROYTOPTOPTSPSPRS. Help her to decrypt it keeping in mind that Eve does not know the initial position of the rotor.

Problem 8. «16QAM»

For sending messages, Alice and Bob use a fiber-optic communication via 16QAM technology. This technology allows to send messages whose alphabet consists of 16 letters, where each letter is usually encoded with a 4-bit Gray code. While a message is transmitted in the channel, single errors in codewords of the Gray code are possible.

Alice has read an interesting book and would like to share her enthusiasm with Bob! Alice sent a short fragment from the book to Bob. Due to the characteristics of the communication channel used, she divided the text into two parts and sent them separately. In the first part, she placed all of the 16 consonants that occurred in this fragment; in the second part, she placed vowels ("y" is a vowel), a **space**, a hyphen and punctuation marks. Then Alice also encoded the letters with Hamming code to be able to correct single errors. She applied a 7-bit Hamming code with the parity-check matrix whose columns are written in lexicographical order.

Bob received the following two parts of ciphertext (given in hexadecimal notation):

Part 1

```
66674C36666F43D3C199900AA1AA325992A
67A59D9B4A8B69330D1BC000153367A5E33
D30E6692D0F349D3321FFFF0ED706667A7F
670D999679F4AA67561BA679B4AA54F34D5
AB0F4AACCF000055CE633670D9DA54CE37F
660DE19CD995335495523CCAAA8F1E03325
86CF48A98CD9B387FD9D546A99E9D200033
3201513FE5B4AA00CCCE9667554CD2CCCB3
330F32A666553CD756AC3E0674E9D369E1D
C6A9999780007F00961E66465519FEA8B25
14CCCB332AA63332CCCE6D2A99AACCCC004
```

Part 2

```
66CA61967319CCD2CE76998CE6433332D19
B46784C65334E999A402ADA0265A99A6633
33319B32D3299698CCC96986619967134CC
B4CE23333334CC6730CE90170CCCD2CE669
996A61999EA63332CCA4C3332D4CD3334CC
D3319994730CCCD3A6669D96A66999699B3
98640CC86CE619676AD4CD3308999866D33
79321C33210B4C6732199B53218019A404C
D2DE65A986663398CCCCCB5319CC6665997
B96A63398CD9CCD2CD9A399A66339866619
98CD9CC325A6339CCE619998C04C66CE633
996A61998CF66967334CC66CA6199865E(0)
```
$_2$

Also, he received the following number sequence: 22, 19, 3, 3, 36, 53, 3, 33, 20, 28. Each number indicates how many consonants are contained between the punctuation marks.

Recover the text and find the main character of the book Alice has read!

Problem 9. «Calculator»

Alice and Bob are practicing in developing toy cryptographic applications for smartphones. This year they have invented `Calculator` that allows one to perform the following operations modulo 2019:

- to insert at most 4-digit positive integers (digits from 0 to 9);
- to perform addition, subtraction and multiplication of two numbers;
- to store temporary results and read them from the memory.

Suppose that Alice wants to send Bob a ciphertext y (given by a 4-digit integer). She sends y from her smartphone to Bob's `Calculator` memory. To decrypt y, Bob needs to get the plaintext x (using his `Calculator`) by the rule $x = f(y) \bmod 2019$, where f is a secret polynomial known to Alice and Bob only.

At the most inopportune moment, Bob dropped his smartphone and broke its screen. Now, the button [+] as well as all digits except [2] are not working.

Help Bob to invent an efficient algorithm how to decrypt any ciphertext y using `Calculator` in his situation if the current secret polynomial is $f(y) = y^5 + 1909y^3 + 401y$. More precisely, suggest a short list of commands, where each command has one of the following types ($1 \leqslant j, k < i$):

$$S_i = y, \qquad S_i = 2, \qquad S_i = 222, \qquad S_i = S_j - S_k,$$
$$S_i = 22, \qquad S_i = 2222, \qquad S_i = S_j * S_k.$$

The first command has to be $S_1 = y$. In the last command, the resulted plaintext x has to be calculated. We remind that all calculations are modulo 2019. In particular, the integer 2222 becomes 203 immediately after entering. The shorter the list of commands you suggest, the more scores you get for this problem.

Example. The following list of commands calculates $x = y^2 - 4$:

Command	Result
$S_1 = y$	y
$S_2 = S_1 * S_1$	y^2
$S_3 = 2$	2
$S_4 = S_3 * S_3$	4
$S_5 = S_2 - S_4$	$y^2 - 4$

Problem 10. «APN + Involutions»

Alice wants to construct a block cipher with heavy use of **involutions** as its subcomponents; this minimizes difference in the algorithms for encryption and decryption. She knows that **almost perfect nonlinear permutations** (**APN permutations**) are the best choice of subcomponents to resist attacks based on differential technique. She wants to construct a set of APN permutations that are involutions for every $n \geqslant 2$.

Alice also knows that any involution can be expressed as the product of disjoint **transpositions**. So, she decides to study the following involution

$$g = \prod_{i=1}^{d} (\alpha_i, {\alpha'}_i),$$

where $\{\alpha_i, {\alpha'}_i\} \cap \{\alpha_j, {\alpha'}_j\} = \emptyset$ for all $i, j \in \{1, ..., d\}$, $i \neq j$, $1 \leqslant d \leqslant 2^{n-1}$.

Alice needs your help to get APN permutations among such involutions g. Find answers to the following questions!

Q1 Let

$$\Lambda(g) = \{\alpha_i \oplus {\alpha'}_i : \ i = 1, ..., d\}, \qquad \widehat{\Lambda}(g) = [\alpha_i \oplus {\alpha'}_i : \ i = 1, ..., d],$$
$$\mathrm{B}(g) = \{x \oplus y : \ \{x, y\} \subseteq \mathrm{FixP}(g), \ x \neq y\}, \quad \widehat{\mathrm{B}}(g) = [x \oplus y : \ \{x, y\} \subseteq \mathrm{FixP}(g), \ x \neq y],$$

where $\mathrm{FixP}(g)$ is the set of all **fixed points** of g, i. e. $\mathrm{FixP}(g) = \{x \in \mathbb{F}_2^n : \ g(x) = x\}$.

Suppose that g is an APN permutation. Get necessary conditions for multisets $\widehat{\Lambda}(g)$, $\widehat{\mathrm{B}}(g)$ and sets $\Lambda(g)$, $\mathrm{B}(g)$. Prove that if your conditions are not satisfied, then g is not an APN permutation.

Q2 Let

$$\mathrm{d}_{a,b}(g) = |\{x \in \mathbb{F}_2^n : \ g(x \oplus a) \oplus g(x) = b\}|, \quad a, b \in \mathbb{F}_2^n.$$

Let g be an involution and APN. Find $\mathrm{d}_{a,a}(g)$ for each nonzero $a \in \mathbb{F}_2^n$.

Q3 Can you get the nontrivial upper bound on $|\mathrm{FixP}(g)|$?

Q4 Let M_n be the set of all n-bit involutions that are APN permutations.

(a) Can you find the cardinality of M_n for $n = 2, 3, 4$?

(b) Can you find the cardinality of M_n for $n = 5$?

(c) **Bonus problem (extra scores, a special prize!)**
Let $n \geqslant 6$. Can you get the lower and the upper bounds for the cardinality of M_n? Can you describe involutions from M_n? Can you suggest constructions for involutions from M_n?

Note that the mapping $x \mapsto x^{-1}$ in the Galois field $GF(2^n)$ belongs to M_n for odd $n \geqslant 3$.

Turn to the next page.

Remark. Let us recall the relevant definitions.

- $\mathbb{F}_2^n$ is the vector space of dimension over $\mathbb{F}_2 = \{0, 1\}$.
- A vector $x \in \mathbb{F}_2^n$ has the form $x = (x_1, ..., x_n)$, where $x_i \in \mathbb{F}_2$. For two vectors $x, y \in \mathbb{F}_2^n$ their sum is $x \oplus y = (x_1 \oplus y_1, ..., x_n \oplus y_n)$, where $\oplus$ stands for XOR operation.
- Let $\widehat{X} = \left[x_1, ..., x_d\right]$ be a multiset with the underlying set $\mathbb{F}_2^n$, where $x_1, ..., x_d \in \mathbb{F}_2^n$.
  Note that all elements in a set are distinct. Unlike a set, a multiset allows for multiple instances for each of its elements.
- A **permutation** s is a mapping from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$ such that $s(x) \neq s(y)$ for all $x, y \in \mathbb{F}_2^n$, $x \neq y$.
- An **involution** s is a permutation that is its own inverse, $s^2(x) = s(s(x)) = x$ for all $x \in \mathbb{F}_2^n$.
- For any different vectors $\alpha, \beta \in \mathbb{F}_2^n$, a permutation s is called a **transposition** if $s(\alpha) = \beta$, $s(\beta) = \alpha$ and $s(x) = x$ for all $x \in \mathbb{F}_2^n \backslash \{\alpha, \beta\}$; it is denoted by $s = (\alpha, \beta)$.
- A permutation s is called **APN** (Almost Perfect Nonlinear) if, for every nonzero $a \in \mathbb{F}_2^n$ and every $b \in \mathbb{F}_2^n$, the equation $s(x \oplus a) \oplus s(x) = b$ has at most 2 solutions.

Problem 11. «Conjecture»

Let $\mathbb{F}_2$ be the finite field with two elements and n be any positive integer larger than or equal to 3. Let $f(X)$ be an irreducible polynomial of degree n over $\mathbb{F}_2$. It is known that the set of the equivalence classes β of polynomials over $\mathbb{F}_2$ modulo $f(X)$ is a finite field of order 2^n, that we shall denote by $\mathbb{F}_{2^n}$. It is known that different choices of the irreducible polynomial give automorphic finite fields and such choice has then no incidence on the algebraic problems on the corresponding fields.

Problem for a special prize! Prove or disprove the following

Conjecture. Let k be co-prime with n. For every $\beta \in \mathbb{F}_{2^n}$, let $F(\beta) = \beta^{4^k-2^k+1}$. Let $\Delta = \{F(\beta) + F(\beta+1) + 1;\ \beta \in \mathbb{F}_{2^n}\}$. For every distinct nonzero v_1, v_2 in $\mathbb{F}_{2^n}$, we have

$$|\{(x,y,z) \in \Delta^3;\ v_1x + v_2y + (v_1+v_2)z = 0\}| = 2^{2n-3}.$$

Example for $n = 3$: we can take $f(X) = X^3+X+1$, then each element β 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 $a_0, a_1, a_2 \in \mathbb{F}_2$. The element 0 corresponds to the null polynomial; and the unity, denoted by 1, corresponds to the constant polynomial 1. We can calculate the table of multiplication in $\mathbb{F}_{2^3}$ (the table of addition just corresponds to adding polynomials of degree at most 2); this allows calculating any power of any element of the field and check the property.