Problem 1. «A secret sharing»

Special Prize from the Program Committee!

Alice, Bob and Caroline are going to create a secret sharing system. They choose some subset $M \subseteq \mathbb{F}_2^n$ and want to share a secret element u from M using the following way: the secret is represented as $x \oplus y \oplus z$ where x, y, z are different elements of $\overline{M} = \mathbb{F}_2^n \backslash M$; Alice, Bob and Caroline will store x, y and z correspondingly. Here $\mathbb{F}_2^n$ is the set of all binary vectors of length n.

To use the system, the sets M and $\overline{M}$ should satisfy the following conditions:

1) each element $u \in M$ can be represented as $u = x \oplus y \oplus z$, where x, y, z are different elements of $\overline{M}$;

2) for all different $x, y, z \in \overline{M}$ it is right $x \oplus y \oplus z \in M$.

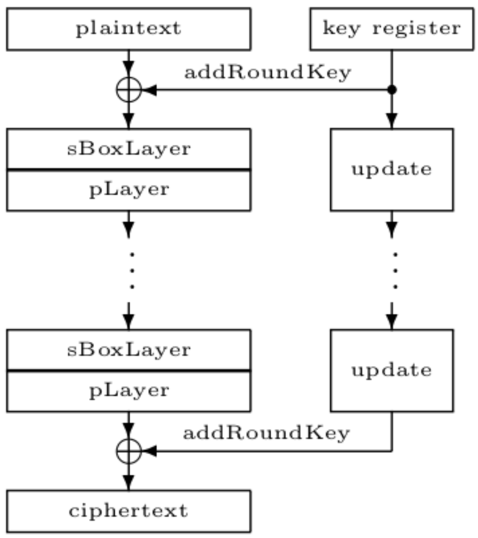
Help them to implement the system suggesting an explicit construction of the set M for an arbitrary n.

Problem 2. «The machine DH-d»

Let G be a cyclic group of a large prime order q and g be a generator of G. Tom designed the machine `DH-d` that on input (g, g^x) outputs g^{x^d}. Here g^x is an arbitrary element of G and d is a small fixed positive integer.

Use the machine `DH-d` to solve the Diffie – Hellman problem, that is, find g^{xy} from (g, g^x, g^y). Suggest a solution with the minimal requests to the machine.

Problem 3. «A modification of PRESENT»



Peter decided to modify the wellknown cipher PRESENT.

At first we give a description of PRESENT according to the paper PRESENT: An Ultra-Lightweight Block Cipher

It is a classical Substitution-Permutation network (SP-network) that consists of 31 rounds with the block size equal to 64 bits and the key size equal to 80 bits. Each of the 31 rounds consists of an XOR operation to introduce a round key K_i for $1 \leqslant i \leqslant 32$, where K_{32} is used for post-whitening, a non-linear substitution layer, and a linear bitwise permutation P. The non-linear layer uses a single 4-bit S-box S which is applied 16 times in parallel in each round.

addRoundKey. Given current `state` $b_{63} \ldots b_0$ and round key $K_i = k^i_{63}k^i_{62} \ldots k^i_0$ for $1 \leqslant i \leqslant 32$, `addRoundKey` consists of the operation $b_j \to b_j \oplus k^i_j$ for $0 \leqslant j \leqslant 63$.

sBoxlayer. The S-box is a permutation from $\mathbb{F}_2^4$ to $\mathbb{F}_2^4$. For `sBoxLayer` the current `state` $b_{63} \ldots b_0$ is considered as sixteen 4-bit words $w_{15} \ldots w_0$ where $w_i = b_{4\cdot i+3}||b_{4\cdot i+2}||b_{4\cdot i+1}||b_{4\cdot i}$ for $0 \leqslant i \leqslant 15$ and the output nibble $S[w_i]$ provides the updated `state` values in the obvious way. The action of this box in hexadecimal notation is given by the following table.

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S(x)$	c	5	6	b	9	0	a	d	3	e	f	8	4	7	1	2

pLayer. The bit permutation is given by the table. Bit i of `state` is moved to bit position $P(i)$.

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$P(i)$	0	16	32	48	1	17	33	49	2	18	34	50	3	19	35	51	4	20	36	52	5	21	37	53	6	22	38	54	7	23	39	55

i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$P(i)$	8	24	40	56	9	25	41	57	10	26	42	58	11	27	43	59	12	28	44	60	13	29	45	61	14	30	46	62	15	31	47	63

The key schedule. The user-supplied key is stored in a key register `K` and represented as $\mathtt{k}_{79}\mathtt{k}_{78} \ldots \mathtt{k}_0$. At round i the 64-bit round key $K_i = k_{63}k_{62} \ldots k_0$ consists of the 64 leftmost bits of the current contents of register `K`. Thus at round i we have that: $K_i = k_{63}k_{62} \ldots k_0 = \mathtt{k}_{79}\mathtt{k}_{78} \ldots \mathtt{k}_{16}$. After extracting the round key K_i, the key register $\mathtt{K} = \mathtt{k}_{79}\mathtt{k}_{78} \ldots \mathtt{k}_0$ is updated as follows. The key register is rotated by 61 bit positions to the left, then the left-most four bits $\mathtt{k}_{79}\mathtt{k}_{78}\mathtt{k}_{77}\mathtt{k}_{76}$ are passed through the PRESENT S-box, and finally the `round_counter` value i is XORed with bits $\mathtt{k}_{19}\mathtt{k}_{18}\mathtt{k}_{17}\mathtt{k}_{16}\mathtt{k}_{15}$ of `K` with the least significant bit of `round_counter` on the right.

see the next page

What Peter has modified:

- In **sBoxlayer**, he changed S-box to the following

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S(x)$	c	8	d	1	e	a	7	b	4	0	5	9	6	2	f	3

- In **pLayer**, he applied permutation P^3 instead of P.
- In **the key schedule**, he rotated the key register by 16 bit positions to the left instead of 61. And he used his new S-box from **sBoxlayer** here.
- Finally, he reduced the number of rounds to 15.

As a result Peter got the new cipher `Peter-PRESENT`. Below you can find examples of test vectors for `Peter-PRESENT` that are given as integers in hexadecimal notation.

plaintext	key	ciphertext
`0000000000000000`	`00000000000000000000`	`f778777b0774f772`
`ffffffffffffffff`	`00000000000000000000`	`888708847883888d`
`0000000000000000`	`ffffffffffffffffffff`	`7ff8fffb0ffc7ffa`
`ffffffffffffffff`	`ffffffffffffffffffff`	`00078004700b0005`

Peter states that his modification is rather good. But his friend Mark does not think so. He claims that it is enough to get only two pairs «plaintext–ciphertext» (P_1, C_1), (P_2, C_2), where $C_i =$ `Peter-PRESENT`(P_i, K), $i = 1, 2$, and K is the unknown key, for reading any message C encrypted with this key K in the ECB mode.

Peter decides to argue with Mark and presents the following pairs, where P_1 and P_2 forms the message `!NSUCRYPTO-2015!` (ASCII codes of letters and little-endian order of bytes are used to form 64-bits integers as the inputs $b_{63}b_{62}\ldots b_0$):

`!NSUCRYP` $\rightarrow$ $P_1 =$ `5059524355534e21` $\rightarrow$ $C_1 =$ `2ddbf038b201448f`

`TO-2015!` $\rightarrow$ $P_2 =$ `21353130322d4f54` $\rightarrow$ $C_2 =$ `d4bf134bd57f4df2`

And he asks Mark to read the secrete message whose ciphertext C is:

```
C =  37aa471c953defe1   91aa595c0236edc9   80f10a020c33e5cb
     ddf14e15923df8dc   8cf8470d027af1db   9caa061e9537ead1
     92e10a1e072ea2c0   d1f1501e9b27f2c3   94e750140134e386
     92f6595b093de3d2   99ec435b0235ebdc   83ef4b099b37f886
     9eef461e4f76eecf   9eaa4912093df8d2   ddf15e129231f8c7
     89ec45184f3ee4cf   94e25e5b9c36eddc   87e55a0b9221a2d2
     ddae471d0e36a2d2   9aec4b159533efca   98e5495b0b34eb86
     9cf643180e34ffc3   89aa4c124f21e4c9   ddf6594ad57aefce
     dbfb500e9b34efc5
```

Can Mark win the argument?

Problem 4. «Guess the cipher»

There is a cipher NSUCRYPTO-2015 that encrypts messages written in 26-letters English alphabet from A to Z. A message length is not more than 50 letters.

Can you recognize what is the cipher algorithm if you can get the ciphertext for any your correct input message here?

Problem 5. «Hypothesis»

Special Prize from the Program Committee!

Prove the following hypothesis or find a counterexample to it.

Hypothesis. For all $n \geqslant 2$ there exists a Boolean function $g : \mathbb{F}_2^{n-1} \to \mathbb{F}_2$ in disjunctive normal form, *where every variable appears not more than one time*, such that a binary sequence $\{u_1, u_2, \ldots\}$ produced for all $t \geqslant 1$ from the initial state $u_1, \ldots, u_n$ by the following rule

$$u_{t+n} = u_t \oplus g(u_{t+1}, u_{t+2}, \ldots, u_{t+n-1})$$

has the maximal possible period equal to 2^n.

Remark I. A Boolean function g in m variables is given in disjunctive normal form if $g(x_1, \ldots, x_m) = A_1 \vee \ldots \vee A_k$, where A_i is a conjunction of variables or their negations, $i = 1, \ldots, k$.

Remark II. In the table, the functions g that confirm the hypothesis for small n are presented.

n	the examples of $g(x_1, \ldots, x_{n-1})$
2	1
3	$x_1 \vee \overline{x}_2$
4	$x_1 \vee \overline{x}_2\overline{x}_3$, $\quad x_1 \vee x_2 \vee \overline{x}_3$
5	$x_2 \vee \overline{x}_1\overline{x}_3\overline{x}_4$, $\quad x_1 \vee x_2x_3 \vee \overline{x}_4$

Problem 6. «A binary tape»

A cipher machine works with a binary infinite tape that starts with an input word of length n and all its other elements are zero. The machine encrypts an input word and writes the result instead of it.

The cipher machine can do two operations:

1) copy any symbol of the tape to other position;
2) apply some fixed one-to-one function $S : \mathbb{F}_2^m \to \mathbb{F}_2^m$ to the first m symbols, where $\mathbb{F}_2 = \{0, 1\}$.

Find the conditions for S such that the machine can perform any bijective mapping of words of length n.

Examples of operations.

1) For instance, the machine can copy the third symbol to the fifth place:

1	1	1	0	**0**	0	1	1	1	...

the result will be

1	1	1	0	1	0	1	1	1	...

2) Let m be 3 and $S(x, y, z) = (x, y, x \oplus z)$; applying S to the first three symbols:

1	**1**	**1**	0	0	0	1	1	1	...

the result will be

1	**1**	**0**	0	0	0	1	1	1	...

Problem 7. «*Palindrome* cipher»

The company *Palindrome* had been using the block cipher `DES` to encrypt its documents for 12 years since the foundation until its engineers took a decision to use the block cipher `Blowfish` in addition to `DES`. It was in 2005 year. So, up to now all its documents are encrypted by `DES` and then the result is also encrypted by `Blowfish`. The ciphering is conducted in EBC mode. Both ciphers `DES` and `Blowfish` have the same key and block lengths equal to 64 bits. The descriptions of these ciphers can be found here: `DES` and `Blowfish`.

As a result of information leakage, that occurred during the celebration of the anniversary of the company, the text of a greeting card leaked to the Internet. The text of the greeting card was

```
Dear colleagues! Congratulations for our wonderful journey of 20 years
of success and we hope the same for the future also!
```

And the ciphertext of that greeting card was

```
C =  83c100497b13525e  fc8d3201d58ab9ed  f6820425912ce184
     23034db7b4408629  4df36ca87ad39f4a  99277e6f1e217dfd
     f2eab13d1161e849  0fe72e9b98fc1e8a  0aa5680e3b4022cb
     4e44c8745afae37f  bd5d6d49292bd1b2  9386f2f383061bfd
     ae8fca32e6745687  565d353f3bbb1204  aa79742f7ab55fb1
     123e6cf37fbad6fe
```

Could you decrypt the following ciphertext that was intercepted in the company network few weeks ago:

```
C =  cf414505b7d3aee3  36f48ae753ec799c  fb49aaea17fa2a38  2992ed164e9622aa
     0b64549dad59a803  0b93be9baf9339e6  fe9780d39168bdff  10d77405d1b51a6a
     5475ddf991ef3ad9  85a6c0c451b75da5  aa4c59ec0c40af09  852b70cebeb127b9
     43c362dccbebf21e  dbb2b086aba67212  1c92e2f327a03b05  b1affd236d8e0f9c
     62386237b27597b4  cbe8ec78b07f4ce6
```

It is known that an encryption 128-bit key is changed dynamically every day according to certain rules and it is always a sequence of 128 bits where each of 16 bytes is given by ASCII codes of figures from `0` to `9`. The first 64 bits form a `DES` key and the other 64 bits form a `Blowfish` key.

see the next page

Here we present some technical information of the company encryption. Below you can find examples of test vectors for combination of both ciphers DES and Blowfish. They are given as 64-bit integers $b_{63}b_{62}\dots b_0$ in hexadecimal notation.

plaintext	DES key	Blowfish key	ciphertext
0000000000000000	0000000000000000	0000000000000000	561543527d054ad0
0000000000000000	0000000000000000	ffffffffffffffff	df27adaec8337f57
0000000000000000	ffffffffffffffff	0000000000000000	11148646af0d82e9
ffffffffffffffff	0000000000000000	ffffffffffffffff	18708bdc3837046f
6c6f632072616544	3837363534333231	3132333435363738	72e66b26309de78c

To form 64-bit integer $b_{63}b_{62}\dots b_0$ each consequent 8 symbols of an original text (or key) are transformed into their ASCII codes and little-endian order of bytes is used.

For example, let us encrypt the message `Dear colleagues!` using the keys `12345678` and `87654321` for DES and Blowfish correspondingly. We divide it into two blocks of 8 symbols `Dear col` and `leagues!` and encrypt them separately:

$$\texttt{Dear col} \rightarrow P_1 = \texttt{6c6f632072616544} \rightarrow \texttt{DES} \rightarrow T_1 = \texttt{cb32b921efe674e5} \rightarrow$$
$$\rightarrow \texttt{Blowfish} \rightarrow C_1 = \texttt{72e66b26309de78c}$$

$$\texttt{leagues!} \rightarrow P_2 = \texttt{217365756761656c} \rightarrow \texttt{DES} \rightarrow T_2 = \texttt{f3d9c5f0cf2e9e8f} \rightarrow$$
$$\rightarrow \texttt{Blowfish} \rightarrow C_2 = \texttt{2d9f9fd83b15ae75}$$

Thus, the ciphertext is `72e66b26309de78c 2d9f9fd83b15ae75`.

Problem 8. «High-nonlinear functions»

One of interesting classes of one-to-one vectorial Boolean functions of the form $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$, where n is even, is the set of functions such that $F^{-1} = F$. Does this class contain a function with nonlinearity not less than $2^{n-1} - 2^{n/2}$?

Remark. Recall several definitions.

- A vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ can be represented as the set of its n coordinate Boolean functions: $F = (f_1, f_2, \ldots, f_n)$, where $f_1, \ldots, f_n : \mathbb{F}_2^n \to \mathbb{F}_2$;
- the Hamming distance $dist(f, g)$ between two Boolean function $f, g : \mathbb{F}_2^n \to \mathbb{F}_2$ is equal to the number of vectors $x \in \mathbb{F}_2^n$ such that $f(x) \neq g(x)$.
- Nonlinearity nl_F of F is equal to

$$\min_{b \in \mathbb{F}_2^n, b \neq 0} \ \min_{a \in \mathbb{F}_2^n, c \in \mathbb{F}_2} dist(b \cdot F, \ell_{a,c})$$

  where $b \cdot F = b_1 f_1 \oplus b_2 f_2 \oplus \ldots \oplus b_n f_n$ and $\ell_{a,c}(x) = a_1 x_1 \oplus a_2 x_2 \oplus \ldots \oplus a_n x_n \oplus c$.

Problem 9. «Covering radius — 2»

In order to protect a new block cipher against some attack based on S-box approximations Alice needs to solve the following problem.

Let $\mathbb{F}_2^n$ be a n-dimensional vector space over the field $\mathbb{F}_2 = \{0,1\}$. Let $n = 2k$, where k is a positive integer. Evaluate the covering radius and describe the metrical complement of the linear subspace spanned by rows of the following $k \times n$ matrix:

$$M = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & \ldots\ldots\ldots & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & \ldots\ldots\ldots & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & \ldots\ldots\ldots & 0 \\ & & & & & & \ddots & \ddots & \ddots & \\ 0 & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & 0 & 1 \quad 1 \quad 1 & 0 \\ 1 & 0 & \ldots & \ldots & \ldots & \ldots & \ldots & 0 & 0 \quad 0 \quad 1 & 1 \end{pmatrix}$$

Remark I. Recall several definitions and notions. A set $L \subseteq \mathbb{F}_2^n$ is called a *linear subspace* if for every $x, y \in L$ the sum $x \oplus y$ is also in L. The *Hamming distance* $d(x,y)$ between vectors $x, y \in \mathbb{F}_2^n$ is defined as the number of positions where they differ, i. e. $d(x,y) = |\{i \mid x_i \neq y_i\}|$. The Hamming distance from a vector y to a subset $X \subseteq \mathbb{F}_2^n$ is defined as $d(y,X) = \min_{x \in X} d(y,x)$. Since the distance between any two vectors is bounded by n, for an arbitrary subset X there exists the number $d(X)$ such that:

- for every $y \in \mathbb{F}_2^n$ it holds $d(y,X) \leqslant d(X)$;
- there exists a vector $z \in \mathbb{F}_2^n$ with $d(z,X) = d(X)$.

This number is called the *covering radius* of X. Set $\widehat{X} = \{z \in \mathbb{F}_2^n \mid d(z,X) = d(X)\}$ is called the *metrical complement* of X.

Remark II. Let us consider several examples:

- Let X consist of a single vector $x \in \mathbb{F}_2^n$. It is easy to see that $d(X) = n$ and $\widehat{X} = \{x \oplus \mathbf{1}\}$, where $\mathbf{1}$ is the all-ones vector;
- Let Y be a ball of radius r centered at x: $Y = \{y \in \mathbb{F}_2^n \mid d(x,y) \leqslant r\}$. One can verify that $d(Y) = n - r$ and $\widehat{Y} = \{x \oplus \mathbf{1}\}$.

Problem 10. «Bigrams»

Users of a some communication system send messages to each other. Every message is written in English. Eve is a malefactor who intercepts messages in this channel and replaces them with new ones. In detail she does the following: intercepts a message, removes all spaces and punctuation marks from it, splits the message into bigrams starting from the beginning. Then she makes several iterations of destruction of the message. The number of iterations is random.

All bigrams are divided into 3 types:

I. Bigram contains only vowels (i. e. `AA, EI, IO, UO, YU, ...`).

II. Bigram contains only consonants (i. e. `BN, TR, LL, PW, SD, ...`).

III. Bigram contains one vowel and one consonant (i. e. `QA, EC, HI, KO, ...`).

On each iteration Eve takes two random bigrams B_1 and B_2 of the different types and removes them from the message, at the same time she adds a new random bigram B_3 of the third type at the beginning of the message. So, if she chooses bigrams of I and II types (II and III; I and III) she will add an arbitrary bigram of III (I; II) type.

For example, the message `CRYPTO TEXT` can be transformed by Eve like this:

`CRYPTO TEXT` → `(CR)(`<u>`YP`</u>`)(TO)(TE)(`<u>`XT`</u>`)` → `(`<u>`OE`</u>`)(`<u>`CR`</u>`)(TO)(TE)` → `(FE)(TO)(TE)`

The question is the following. You know that Alice has send to Bob the message

`THE MEETING WILL TAKE PLACE AT THREE IN ‘‘EEYORE-EAGLE-BEE CREEK INN’’`

that was intercepted by Eve. She had repeated iterations of destruction until the only one bigram left. Could it be a bigram consisting of one vowel and one consonant?