Problem 1. «A digital signature»

Alice uses a new digital signature algorithm, that turns text a message M into a pair (M, s), where s is an integer and generated in the following way:

1. The special function h transforms M into a big positive integer $r = h(M)$.

2. The number $t = r^2$ is calculated, where $t = \overline{t_1 t_2 \dots t_n}$.

3. The signature s is calculated as $s = t_1 + t_2 + \dots + t_n$.

Bob obtained the signed message

(`Congratulations on the fifth year anniversary of NSUCRYPTO!`, 2018)

from Alice and immediately recognized that something was wrong with the signature! How did he discover it?

A remark. By $t = \overline{t_1 t_2 \dots t_n}$ we mean that $t_1, t_2, \dots, t_n$ are decimal digits and all digits over the bar form decimal number t.

Problem 2. «Orthogonal arrays»

Special Prize from the Program Committee!

Orthogonal arrays are closely connected with cryptographic Boolean functions. Namely, supports of correlation immune functions give orthogonal arrays when their elements are written as the rows of an array.

Given three positive integers n, t and λ such that $t < n$, we call t–$(2, n, \lambda)$ *orthogonal array* every $\lambda 2^t \times n$ binary array (i.e. matrix over the 2-element field) such that, in every subset of t columns of the array, every (binary) t-tuple appears in exactly λ rows. t is called the strength of this orthogonal array.

Find a 4–$(2, 11, \lambda)$ orthogonal array with minimal value of λ.

Problem 3. «Hash function FNV-1a»

Hash function FNV-1a 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 $i = 1, 2, \ldots, n$: $h \leftarrow (h \oplus x_i)g \bmod 2^{128}$;

3) return h.

Here $h_0 = 144066263297769815596495629667062367629$, $g = 2^{88}+315$. The expression $h \oplus x_i$ means that the least significant byte of h is added bitwise modulo 2 with the byte x_i.

Find a collision, that is, two different messages x and x' such that FNV-1a$(x) =$ FNV-1a(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 4. «TwinPeaks2»

Bob realized that his last year cipher `TwinPeaks` (NSUCRYPTO-2017) is not secure enough and modified it. He considerably increased the number of rounds and made rounds more complicated. 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 48 times:

$$(a, b, c, d) \leftarrow (b, c, d, a \oplus S_3(S_1(b) \oplus S_2(b \wedge \neg c \oplus c \vee d) \oplus S_1(d))),$$

Here S_1, S_2, S_3 are secret permutations over 32-bit words; $\neg$, $\wedge$, $\vee$, $\oplus$ are binary bitwise "NOT', "OR", "AND", "XOR" respectively (the operations are listed in descending order of priority). 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

$$Y = \texttt{DEB239852F1B47B005FB390120314478}$$

and captured also Bob's smartphone with the `TwinPeaks2` implementation! Here it is. Now Cooper (and you too) can encrypt any messages with `TwinPeaks2` but still can not decrypt any one.

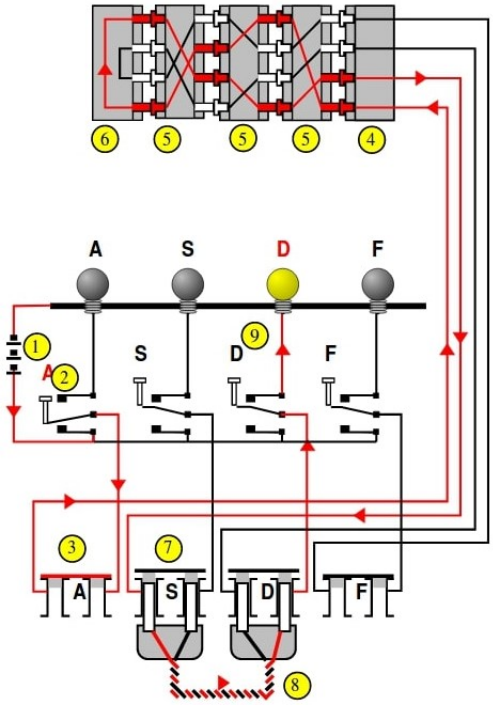
Help Cooper to decrypt Y.

Remark. The ciphertext is given in hexadecimal notation, the first byte is `DE`.

Problem 5. «An Enigmatic Challenge»

The Enigma machine is a symmetric cipher famous for being during the Second World War by the German military. Its internal structure comprises an 26-letter Latin alphabetic permutation, implemented as rotors. The machine used for this problem consists of 3 **rotors** and a **reflector**.

The figure shows how a simplified Enigma machine works. The key components are the set of input switches (2) – which are reduced to 4 in the example but could have been 26 for the Latin alphabet – an input plugboard (3,7,8), three rotors (5), the reflector (6) and the output board (9).



The components have the following functionality:

• **Rotors:** a rotor (5) is a wheel with the upper-case alphabet in order on the rim and a hole for an axle. On both sides of a rotor are 26 electrical contacts each under a letter. Each contact on one side is wired to a contact on the other side at a different position. The rotor implements an one-to-one and onto function between the upper-case letters, where each letter is mapped to a different one (an irreflexive permutation).

• **Reflector:** the reflector (6) is positioned after the rotors and has contacts for each letter of the alphabet on one side only. The letters are wired up in pairs, so that an input current on a letter is reflected back to a different letter.

The input message: is permuted by the rotors, passes through the reflector and then goes back through the rotors in the reverse order (as depicted in the figure). Finally, the light bulb indicates the encrypted letter. The plugboard plays no role in permuting the letter for this challenge, although it could have.

Turn to the next page.

To prevent simple frequency analysis attack the right rotor rotates with every new input. After the right rotor completed a full rotation (after 26 letters were encrypted), the middle rotor rotates once. Similarly, after the middle rotor completes a full rotation (and the right rotor complete 676 rotations), the left rotor rotates once.

Challenge: you will play the role of an attacker that knows the source of the plaintext to be encrypted. You are given a ciphertext corresponding to a plaintext taken from this known source which happens to be "Moby Dick" by Herman Melville, and you are asked to recover the plaintext. The plaintext consists only of trimmed capital letters with no punctuation marks and spaces and is contiguous. All letters are from the Latin alphabet. Extra information on the settings of the rotors is provided: the configuration of the first rotor is very close to the one used in the 1930 commercial version: `EKMFLGDQVZNTOWYHXUSPAIBRCJ`.

Ciphertext:

```
RHSM ZHXX AOWW ZTWQ QQMB CRZA BARN MLAV MLSX SPBA ZTHG
YLGE VGZG KULJ FLOZ RQAW YGAA DCJB YWBW IYQQ FAAO RAGK
BGSW OARG EYSP IKYE LLUO YCNH HDBV AFKD HETA ONNR HXHE
BBRT ROZD XJCC OMXR PNSW UAZB TNJY BANH FGCS GJWY YTBV
VGLX KUZW PARO NMXP LDLZ ICBK XVSJ NXCF SOTA AQYS YZFX
MZDH MSZI ABAH RFXT FTPU VWMC PEXQ NZVA LMFX BHKG QGYS
BIYE MEUE PJNR AVTL JSUZ PLHQ MOUI IQFD HVXI NOOJ YJAF
 WAVU PVQA FMKP AHLK XJYD GITB QSPK CUZU XPRK MUJJ YRJ
```

Link to "Moby Dick" text file: click here.

Problem 6. «Sylvester matrices»

Special Prize from the Program Committee!

Consider two univariate polynomials over the 2-element field, $P_1(x)$ of degree m and $P_2(x)$ of degree n where $P_1(x) = a_m x^m + \ldots + a_0$ and $P_2(x) = b_n x^n + \ldots + b_0$. The *Sylvester matrix* is an $(m+n) \times (m+n)$ matrix formed by filling the matrix beginning with the upper left corner with the coefficients of $P_1(x)$, then shifting down one row and one column to the right and filling in the coefficients starting there until they hit the right side. The process is then repeated for the coefficients of $P_2(x)$. All the other positions are filled with zero.

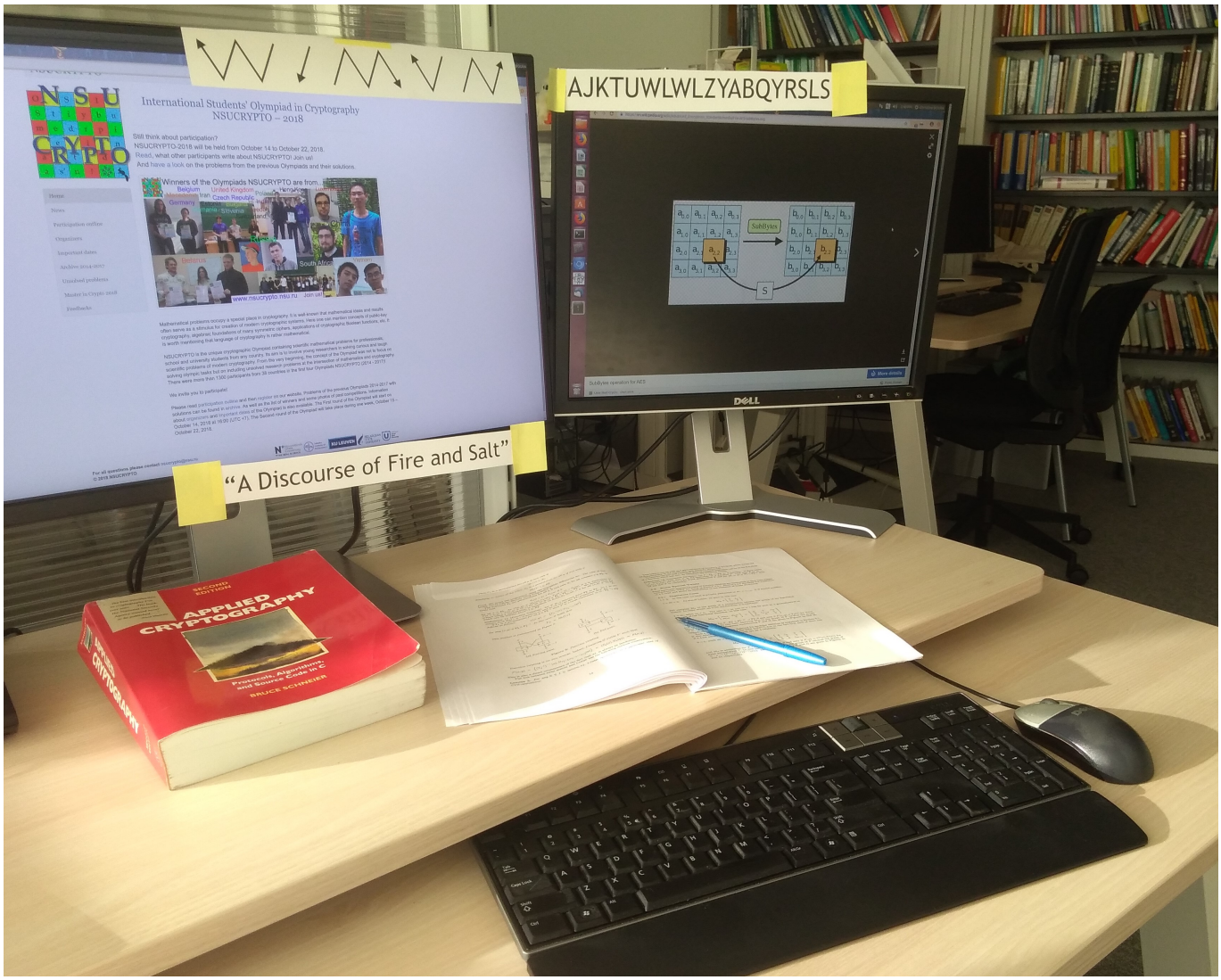
Let $n > 0$, $m > 0$. Prove whether there exist $(m+n) \times (m+n)$ invertible Sylvester matrices whose inverse are Sylvester matrices as well.

Example. For $m = 4$ and $n = 3$, the Sylvester matrix is the following:

$$\begin{pmatrix} a_4 & a_3 & a_2 & a_1 & a_0 & 0 & 0 \\ 0 & a_4 & a_3 & a_2 & a_1 & a_0 & 0 \\ 0 & 0 & a_4 & a_3 & a_2 & a_1 & a_0 \\ b_3 & b_2 & b_1 & b_0 & 0 & 0 & 0 \\ 0 & b_3 & b_2 & b_1 & b_0 & 0 & 0 \\ 0 & 0 & b_3 & b_2 & b_1 & b_0 & 0 \\ 0 & 0 & 0 & b_3 & b_2 & b_1 & b_0 \end{pmatrix}$$

Problem 7. «Stickers»

Bob always takes into account all the recommendations of security experts. He switched from short passwords to long passphrases and changes them every month. Bob usually chooses passphrases from the books he is reading. Passphrases are so lengthy and are changed so often! In order to not forget them, Bob decided to use stickers with hints. He places them on his monitors (ooh, experts...). The only hope is that Bob's hint system is reliable because it uses encryption. But is that true? Could you recover Bob's current passphrase from the photo of his workspace?



Problem 8. «Bash-S3»

The sponge function Bash-f uses the permutation $S3$ that transforms a triple of 64-bit binary words a, b, c in the following way:

$$S3(a, b, c) = (b \vee \neg c \oplus a,\ \ a \vee c \oplus b,\ \ a \wedge b \oplus c).$$

Here $\neg$, $\wedge$, $\vee$, $\oplus$ denote the binary bitwise operations "NOT", "OR", "AND", "XOR" respectively. The operations are listed in descending order of priority. Let w^k also denote the cyclic shift of a 64-bit word w to the left by $k \in \{1, 2, \ldots, 63\}$ positions.

Alice wants to strengthen $S3$. She can add by XOR any input a, b, c or its cyclic shift to any output. She must use at least one cyclic shift and she cannot add two identical terms to the same output.

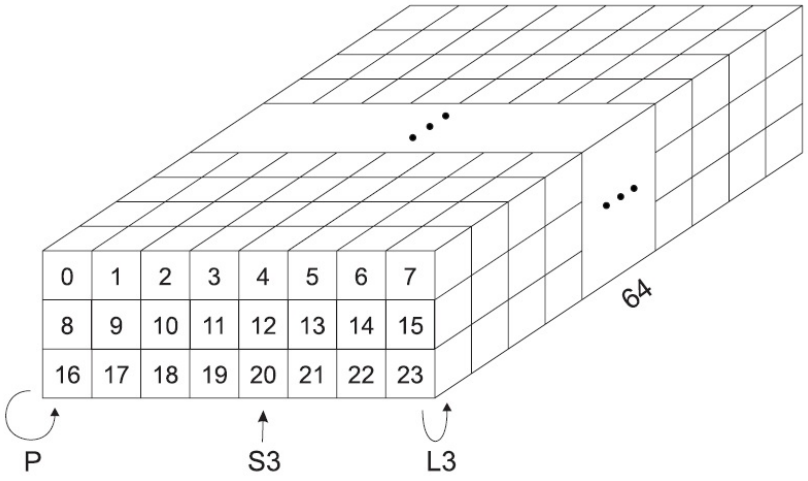
Help Alice to change $S3$ in such a way that a modified $S3$ will be still a permutation!

Remark 1. The descending order of operation priority means that, for example, in the expression $b \vee \neg c \oplus a$, we firstly calculate $\neg c$, then calculate $b \vee \neg c$, and after that the final result.

Remark 2. The modification

$$(b \vee \neg c \oplus a \oplus a^{11},\ \ a \vee c \oplus a^{7} \oplus c,\ \ a \wedge b \oplus b^{32})$$

is allowed but it does not satisfy the permutation condition.



Problem 9. «Metrical cryptosystem – 2»

Let $\mathbb{F}_2^n$ be an n-dimensional vector space over the field $\mathbb{F}_2 = \{0, 1\}$. Alice and Bob exchange messages using the following cryptosystem.

- First, they use a supercomputer to calculate two special large secret sets $A, B \subseteq \mathbb{F}_2^n$ which have the following property: there exists a constant ℓ ($\ell \geqslant 26$), such that for any $x \in \mathbb{F}_2^n$ it holds
$$d(x, A) + d(x, B) = \ell,$$
where $d(x, A)$ denotes Hamming distance from the vector x to the set A.

- Alice then saves the number ℓ, the set A and a set of vectors $a_1, a_2, \ldots, a_r$ such that for any $k : 0 \leqslant k \leqslant \ell$, there is a vector a_i at distance k from A. Similarly, Bob saves the number ℓ, the set B and a set of vectors $b_1, b_2, \ldots, b_s$ such that for any $k : 0 \leqslant k \leqslant \ell$, there is a vector b_i at distance k from B.

- Text messages are encrypted letter by letter. In order to encrypt a letter Alice replaces it with its number in the alphabet, say k. Then she chooses some vector a_i at distance k from the set A and sends this vector over to Bob. Bob then calculates the distance $d(a_i, B)$ and using the property of the sets A, B, calculates $k = \ell - d(a_i, B)$. So, he gets the letter Alice sent. If Bob wants to send an encrypted message to Alice, he does the same but using his saved vectors and the set B.

Eve was able to hack the supercomputer when it was calculating the sets A and B. She extracted the set C from its memory, which consists of all vectors of $\mathbb{F}_2^n$ that are at distance 1 or less from either A or B. She also learned that ℓ is even.

Help Eve to crack the presented cryptosystem (to decrypt any short intercepted message)! You know that she has an (illegal) access to the supercomputer, which can calculate and output the list of distances from all vectors of $\mathbb{F}_2^n$ to any input set D in reasonable (but not negligible) time.

Remark I. Recall several definitions and notions. The *Hamming distance* $d(x, y)$ between vectors x and y is the number of coordinates in which these vectors differ. Distance from vector $y \in \mathbb{F}_2^n$ to the set $X \subseteq \mathbb{F}_2^n$ is defined as $d(y, X) = \min_{x \in X} d(y, x)$.

Problem 10. «A fixed element»

A polynomial $f(X_1, \ldots, X_n) \in \mathbb{F}_2[X_1, \ldots, X_n]$ is called *reduced* if the degree of each X_i in f is at most 1. For $0 \leqslant r \leqslant n$, the rth order Reed — Muller code of length 2^n, denoted by $R(r, n)$, is the $\mathbb{F}_2$-space of all reduced polynomials in $X_1, \ldots, X_n$ of total degree $\leqslant r$. We also define $R(-1, n) = \{0\}$.

The general linear group $\mathrm{GL}(n, \mathbb{F}_2)$ acts on $R(r, n)$ naturally: Given $A \in \mathrm{GL}(n, \mathbb{F}_2)$ and $f(X_1, \ldots, X_n) \in R(r, n)$, Af is defined to be the reduced polynomial obtained from $f((X_1, \ldots, X_n)A)$ by replacing each power X_i^k $(k \geqslant 2)$ with X_i. Consequently, $\mathrm{GL}(n, \mathbb{F}_2)$ acts on the quotient space $R(r, n)/R(r-1, n)$.

Let $A \in \mathrm{GL}(n, \mathbb{F}_2)$ be such that its characteristic polynomial is a primitive irreducible polynomial over $\mathbb{F}_2$. Prove that the only element in $R(r, n)/R(r-1, n)$, where $0 < r < n$, fixed by the action of A is 0.

Problem 11. «Disjunct Matrices»

Special Prize from the Program Committee!

Disjunct Matrices are used in some key distribution protocols for traitor tracing. Disjunct Matrices (DM) are a particular kind of binary matrices which have been especially applied to solve the Non-Adaptive Group Testing (NAGT) problem, where the task is to detect any configuration of t defectives out of a population of N items. Traditionally, the methods used to construct DM leverage on error-correcting codes and other related algebraic techniques.

Let $A = (x_1^\top, x_2^\top, \ldots, x_N^\top)$ be an $M \times N$ binary matrix. Then, A is called *t-disjunct* if, for all subsets of t columns $S = \{x_{i_1}, \ldots, x_{i_t}\}$, and for all remaining columns $x_j \notin S$, it holds that

$$\operatorname{supp}(x_j) \not\subseteq \bigcup_{k=1}^{t} \operatorname{supp}(x_{i_k}),$$

where $\operatorname{supp}(x)$ denotes the set of coordinate positions of a binary vector x with 1s. In other words, a matrix A is t-disjunct if for every subset S of t columns the support of any other column is not contained in the union of the supports of the columns in S.

Prove what is the minimum number of rows in a 5-disjunct matrix.