On the Sixth International Olympiad in Cryptography
NSUCRYPTO*

A. Gorodilova¹, N. Tokareva¹,², S. Agievich³, C. Carlet⁴, E. Gorkunov¹,⁵, V. Idrisova¹, N. Kolomeec¹, A. Kutsenko¹,⁵, R. Lebedev⁵, S. Nikova⁶, A. Oblaukhov¹, I. Pankratova⁷, M. Pudovkina⁸, V. Rijmen⁶, A. Udovenko⁹

¹Sobolev Institute of Mathematics, Novosibirsk, Russia
²Laboratory of Cryptography JetBrains Research
³Belarusian State University, Minsk, Belarus
⁴University of Paris 8, Paris, France
⁵Novosibirsk State University, Novosibirsk, Russia
⁶ESAT-COSIC, KU Leuven, Leuven, Belgium
⁷Tomsk State University, Tomsk, Russia
⁸Bauman Moscow State Technical University, Moscow, Russia
⁹SnT, University of Luxembourg, Esch-sur-Alzette, Luxembourg

E-mail: nsucrypto@nsu.ru

Abstract. NSUCRYPTO is the unique cryptographic Olympiad containing scientific mathematical problems for professionals, school and university students from any country. Its aim is to involve young researchers in solving curious and tough scientific problems of modern cryptography. From the very beginning, the concept of the Olympiad was not to focus on solving olympic tasks but on including unsolved research problems at the intersection of mathematics and cryptography. The Olympiad history starts in 2014. In 2019, it was held for the sixth time. In this paper, problems and their solutions of the Sixth International Olympiad in cryptography NSUCRYPTO’2019 are presented. We consider problems related to attacks on ciphers and hash functions, protocols, Boolean functions, Dickson polynomials, prime numbers, rotor machines, etc. We discuss several open problems on mathematical countermeasures to side-channel attacks, APN involutions, S-boxes, etc. The problem of finding a collision for the hash function Curl27 was partially solved during the Olympiad.

Keywords. cryptography, ciphers, hash functions, Hamming code, slide attack, threshold implementation, Dickson polynomial, APN function, Olympiad, NSUCRYPTO.

*The work of the first two authors and the sixth author was supported by Mathematical Center in Akademgorodok under agreement No. 075-15-2019-1613 with the Ministry of Science and Higher Education of the Russian Federation and Laboratory of Cryptography JetBrains Research. The work of the seventh, eighth and eleventh authors was supported by Russian Foundation for Basic Research (projects no. 20-31-70043, 18-07-01394, 19-31-90093).
Introduction

NSUCRYPTO (Non-Stop University Crypto) is the International Olympiad in cryptography that was held for the sixth time in 2019.

Interest in the Olympiad around the world is significant. This year, there were hundreds of participants from 26 countries; 42 participants in the first round and 21 teams in the second round from 16 countries were awarded with prizes and honorable diplomas. The Olympiad program committee includes specialists from Belgium, France, the Netherlands, the USA, Norway, India, Luxembourg, Belarus', Kazakhstan, and Russia.

Let us shortly formulate the format of the Olympiad. One of the Olympiad main ideas is that everyone can participate! Each participant chooses his/her category when registering on the Olympiad website [15]. There are three categories: “school students” (for junior researchers: pupils and high school students), “university students” (for participants who are currently studying at universities) and “professionals” (for participants who have already completed education or just want to be in the restriction-free category). Awarding of the winners is held in each category separately.

The Olympiad consists of two independent Internet rounds: the first one is individual (duration 4 hours 30 minutes) while the second round is a team one (duration 1 week). The first round is divided into two sections: A — for “school students”, B — for “university students” and “professionals”. The second round is common to all participants. Participants read the Olympiad problems and submit their solutions using the Olympiad website. The language of the Olympiad is English.

The Olympiad participants are always interested in solving different problems of various complexities at the intersection of mathematics and cryptography. They show their knowledge, creativity and professionalism. That is why the Olympiad not only includes interesting tasks with known solutions but also offers unsolved problems in this area. This year, one of such open problems, “Curl27” (see section 2.14), was partially solved during the second round! All the open problems stated during the Olympiad history can be found here [16]. On the website we also mark the current status of each problem. For example, in addition to “Curl27”, the problem “Sylvester matrices” was solved by three teams in 2018, the problem “Algebraic immunity” was completely solved during the Olympiad in 2016. And what is important for us, some participants were trying to find solutions after the Olympiad was over. For example, a partial solution for the problem “A secret sharing” (2014) was proposed in [9]. We invite everybody who has ideas on how to solve the problems to send your solutions to us!

The paper is organized as follows. We start with problem structure of the Olympiad in section 1. Then we present formulations of all the problems stated during the Olympiad and give their detailed solutions in section 2. Finally, we publish the lists of NSUCRYPTO’2019 winners in section 3.

Mathematical problems and their solutions of the previous International Olympiads in cryptography NSUCRYPTO from 2014 to 2018 can be found in [2], [1], [14], [10], and [11] respectively.

1 Problem structure of the Olympiad

There were 16 problems stated during the Olympiad, some of them were included in both rounds (Tables 1, 2). Section A of the first round consisted of six problems, whereas the section B contained
seven problems. Three problems were common for both sections. The second round was composed of eleven problems. Five problems of the second round included unsolved questions (awarded special prizes from the Program Committee).

| N | Problem title          | Maximum scores |
|---|------------------------|-----------------|
| 1 | A 1024-bit key         | 4               |
| 2 | The magnetic storm     | 4               |
| 3 | Autumn leaves          | 4               |
| 4 | A rotor machine        | 4               |
| 5 | Broken Calculator      | 4               |
| 6 | A promise              | 6               |

Section A

Table 2: Problems of the second round

| N | Problem title                        | Maximum scores                          |
|---|--------------------------------------|-----------------------------------------|
| 1 | A 1024-bit key                       | 4                                       |
| 2 | Sharing                              | 6 + additional scores for open questions|
| 3 | Factoring in 2019                    | 8                                       |
| 4 | TwinPeaks-3                          | 8                                       |
| 5 | Curl127                              | 10 + additional scores for open questions|
| 6 | 8-bit S-box                          | Unlimited (open problem)                |
| 7 | A rotor machine                      | 4                                       |
| 8 | 16QAM                                | 8                                       |
| 9 | Calculator                           | 6                                       |
| 10| APN + Involutions (extended)         | 12 + additional scores for open questions|
| 11| Conjecture                           | Unlimited (open problem)                |

2 Problems and their solutions

In this section, we formulate all the problems of NSUCRYPTO’2019 and present their detailed solutions paying attention to solutions proposed by the participants.

2.1 Problem “A 1024-bit key”

2.1.1 Formulation

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. 110010111011... turns to 110011001001011...
2.1.2 Solution

Let us encode the binary vector of the key as the corresponding decimal number. It is obvious that this number will increase on the next day, since all the bits on the left from the sequence are not changing, but the first bit of the sequence turns from 0 to 1. Let us note that this number can not increase infinitely since the size of the key is restricted by 1024 bits, so, in the very end the key will be maximal possible and, thus, will consist of all 1s.

Almost all the participants successfully solved the problem.

2.2 Problem “The magnetic storm”

2.2.1 Formulation

A hardware random number generator is a device that generates random sequences consisting of 0s and 1s. Unfortunately, a disturbance caused by a magnetic storm affected this random number generator. As a result, the device had generated a sequence of 0s of length \( k \) (where \( k \) is a positive integer), and then started to generate an infinite sequence of 1s.

Prove that at some point the generator will produce a number \( 1 \ldots 10 \ldots 0 \) that is divisible by 2019.

2.2.2 Solution

Let us prove that a number of form \( 1 \ldots 11 \ldots 1 \) is divisible by 2019. Consider all numbers that consists only of 1s, since there are infinite quantity of these numbers, there can be found a pair of numbers A and B such that they have the same remainder when divided by 2019. Therefore, \( C = A - B = 1 \ldots 10 \ldots 0 \) consisting of \( m \) 1s for some natural \( m \) is divisible by 2019, and, since 2019 is not divisible by 2 and 5, \( C' = C \times 10 \ldots 0 = 1 \ldots 10 \ldots 0 \) is divisible by 2019 for any number of 0s.

There were a lot of correct solutions from the participants.
2.3 Problem “Autumn leaves”

2.3.1 Formulation

Read a hidden message!..

2.3.2 Solution

We see different leaves and spaces between them. It looks like a simple substitution cipher was used there and distinct leaves corresponded to distinct English letters. By English grammar, we can suppose that the second and the third words are “is a”. Then the first word starts with “a” and by its structure can be “autumn” (which is very likely as the autumn landscape is depicted). Also, the leaf 🍁 is the most common letter in the text and we can guess that it is “e”. Then we see “*ea*” in the third line that seems to be “leaf”. As a result the last word becomes “fl**e**” that is “flower”. Finally, we get “Autumn is a second spring when every leaf is a flower” that is a famous quote by Albert Camus. Almost all the participants read the message.
2.4 Problem “A rotor machine”

2.4.1 Formulation

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.

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

2.4.2 Solution

To solve the problem and decrypt the message, one needs to correctly understand the scheme of work. A key for the cipher is the initial position of the rotor. We denote it by a color of the circle on the input side of the rotor that corresponds to the letter O. Table 3 represents the encryption tables depending on the key.

Trying all six possible keys, we find the only one meaningful message POST TO TOP OOPS SORRY STOP ROTOR that corresponds to the “yellow” key.

Almost all the participants solved the problem. The most interesting solutions were obtained by creating real models for this rotor machine, for example by a school student Varvara Lebedinskaya.
Table 3: Encryption tables

|      | O | P | R | S | T | Y |
|------|---|---|---|---|---|---|
| red  | T | Y | S | R | O | P |
| white| R | S | O | P | Y | T |
| purple| Y | R | P | T | S | O |
| green| S | R | P | O | Y | T |
| yellow| S | T | Y | O | P | R |
| blue  | R | T | O | Y | P | S |

(The Specialized Educational Scientific Center of Novosibirsk State University), by the team of Kristina Geut, Sergey Titov, and Dmitry Ananichev (Ural State University of Railway Transport).

2.5 Problem “Broken Calculator”

2.5.1 Formulation

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 (that is to get the result as the remainder of division by 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: \( x \) is equal to the remainder of dividing \( f(y) = y^5 + 1909y^3 + 401y \) by 2019.

At the most inopportune moment, Bob dropped his smartphone and broke its screen. Now, the button \( + \) as well as all digits except \( 1 \) and \( 5 \) are not working.

Help Bob to invent an efficient algorithm how to decrypt any ciphertext \( y \) using Calculator in his situation. More precisely, suggest a short list of commands, where each command has one of the following types \((1 \leq j, k < i)\):

\[
S_i = y, \quad S_i = a, \quad S_i = S_j - S_k, \quad S_i = S_j \ast S_k,
\]

where \( a \) is an at most 4-digit integer consisting of digits 1 and 5 only; for example, \( a = 1, a = 15, a = 551, a = 5115 \), etc.

The first command has to be \( S_1 = y \). In the last command, the resulting plaintext \( x \) has to be calculated. We remind that all calculations are modulo 2019. In particular, the integer 2500 becomes 481 and \(-1000\) becomes 1019 immediately after entering or calculations. The shorter the list of commands you suggest, the more scores you get for this problem.

Example. The following list of commands calculates \( x = y^2 - 55 \):

| Command | Result   |
|---------|----------|
| \( S_1 = y \) | \( y \)   |
| \( S_2 = S_1 \ast S_1 \) | \( y^2 \) |
| \( S_3 = 11 \) | 11       |
| \( S_4 = 5 \) | 5        |
| \( S_5 = S_3 \ast S_4 \) | 55       |
| \( S_6 = S_2 - S_5 \) | \( y^2 - 55 \) |
2.5.2 Solution

Let us present the original solution by the programm committee that has 14 steps.

Let \( a \equiv_m b \) mean that integers \( a \) and \( b \) are congruent modulo \( m \). The following relations hold:

\[
\begin{align*}
f(y) & \equiv_{2019} y^5 + 1909y^3 + 401y \\
& \equiv_{2019} y(y^4 - 110y^2 + 401) \\
& \equiv_{2019} y(y^4 - 2 \cdot 55y^2 + 55^2 - 55^2 + 401) \\
& \equiv_{2019} y((y^2 - 55)^2 - 55^2 + 5 \cdot 22^2) \\
& \equiv_{2019} y((y^2 - 55)^2 - 11^2 \cdot (5^2 - 5 \cdot 2^2)) \\
& \equiv_{2019} y((y^2 - 55)^2 - 11^2 \cdot 5) \\
& \equiv_{2019} y((y^2 - 55)^2 - 11 \cdot 55).
\end{align*}
\]

Thus, the reminder of division of \( f(y) \) by 2019 can be calculated for any \( y \) by the list of commands given in Table 4. A similar solution was found by Borislav Kirilov (Bulgaria, The First Private Mathematical Gymnasium).

| Command    | Result   | Command    | Result   | Command    | Result   |
|------------|----------|------------|----------|------------|----------|
| \( S_1 = y \) | \( y \)    | \( S_4 = S_2 - S_3 \) | \( y^2 - 55 \)     | \( S_7 = S_3 \cdot S_6 \) | \( 11 \cdot 55 \)     |
| \( S_2 = S_1 \cdot S_1 \) | \( y^2 \)    | \( S_5 = S_4 \cdot S_4 \) | \((y^2 - 55)^2 \)    | \( S_8 = S_5 - S_7 \) | \((y^2 - 55)^2 - 11 \cdot 55 \)     |
| \( S_3 = 55 \) | \( 55 \)   | \( S_6 = 11 \) | \( 11 \)     | \( S_9 = S_1 \cdot S_8 \) | \( y((y^2 - 55)^2 - 11 \cdot 55) \)     |

Note. The polynomial \( f(y) = y^5 + 1909y^3 + 401y \) is the Dickson polynomial \( D_5(y, a) = y^5 - 5y^3a + 5ya^2 \) for \( a = 22 \) with coefficients taken modulo 2019.

2.6 Problem “Calculator”

2.6.1 Formulation

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 \( x = f(y) \mod 2019 \), 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 \( 1 \leq j, k < i \):
\[ S_i = y, \quad S_i = 2, \quad S_i = 22, \quad S_i = S_j - S_k, \]
\[ S_i = 22, \quad S_i = 2222, \quad S_i = S_j * S_k. \]

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 |
| \( S_5 = S_2 - S_4 \) | \( y^2 - 4 \) |

### 2.6.2 Solution

The polynomial \( f(y) = y^5 + 1909y^3 + 401y \) is the Dickson polynomial \( D_5(y, a) = y^5 - 5y^3a + 5ya^2 \) for \( a = 22 \) with coefficients taken modulo 2019. The following relations hold:

\[
D_5(y, a) = yD_4(y, a) - aD_3(y, a)
= yD_4(D_2(y, a), a^2) - aD_3(y, a)
= y((y^2 - 2a)^2 - 2a^2) - ay(y^2 - 2a - a).
\]

For \( a = 22 \), the value \( f(y) \) can be calculated for any \( y \) by the list of commands given in Table 5.

#### Table 5: List of commands for the programm committee solution

| Command | Result | Command | Result |
|---------|--------|---------|--------|
| \( S_1 = y \) | \( y \) | \( S_8 = S_7 * S_7 \) | \( (y^2 - 2a)^2 \) |
| \( S_2 = 2 \) | 2 | \( S_9 = S_8 - S_5 \) | \( (y^2 - 2a)^2 - 2a^2 \) |
| \( S_3 = 22 \) | \( a \) | \( S_{10} = S_1 * S_9 \) | \( y((y^2 - 2a)^2 - 2a^2) \) |
| \( S_4 = S_2 * S_3 \) | \( 2a \) | \( S_{11} = S_7 - S_2 \) | \( y^2 - 2a - a \) |
| \( S_5 = S_3 * S_4 \) | \( 2a^2 \) | \( S_{12} = S_1 * S_{11} \) | \( y(y^2 - 2a - a) \) |
| \( S_6 = S_1 * S_1 \) | \( y^2 \) | \( S_{13} = S_3 * S_{12} \) | \( ay(y^2 - 2a - a) \) |
| \( S_7 = S_6 - S_4 \) | \( y^2 - 2a \) | \( S_{14} = S_{10} - S_{13} \) | \( f(y) \) |

What was surprising that the participants found two solutions that has 11 and 13 steps! These solutions were awarded by additional points. The solution with 11 steps were found by Madalina Bolboceanu (Romania, Bitdefender) during the first round (Table 6). The solution with 13 steps were given by Henning Seidler and Katja Stumpp team (Germany, TU Berlin) during the second round. Both of the solution were based on the representation \( f(y) = y((y^2 - 44)(y^2 - 66) - 22^2) \).

### 2.7 Problem “A promise”

#### 2.7.1 Formulation

Young cryptographers, Alice, Bob and Carol, are interested in quantum computings and really want to buy a quantum computer. A millionaire gave them a certain amount of money (say, \( X_A \)
for Alice, $X_B$ for Bob, and $X_C$ for Carol). He also made them promise that they would not tell anyone, including each other, how much money everyone of them had received.

- Could you help the cryptographers to invent an algorithm how to find out (without breaking the promise) whether the total amount of money they have, $X_A + X_B + X_C$, is enough to buy a quantum computer?
- What weaknesses does your algorithm have (if someone breaks the promise)? Does it always protect the secret of the honest participants from the dishonest ones?

### 2.7.2 Solution

This problem is a particular case for the problem “A promise and money” for only three participants (see section 2.8).

### 2.8 Problem “A promise and money”

#### 2.8.1 Formulation

A group of young cryptographers are interested in quantum computational and really want to buy a quantum computer. A millionaire gave them a certain amount of money (say, $n$ cryptographers; $X_i$ for each of them, $i = 1, \ldots, n$). He also made a promise from them that they would not tell anyone, including each other, how much money everyone of them had received.

- Could you help the cryptographers to invent an algorithm how to find out (without breaking the promise) whether the total amount of money they have, $\sum_{i=1}^{n} X_i$, is enough to buy a quantum computer?
- What do you think whether there are such algorithms protecting the secrets of honest participants from dishonest ones?
- What weaknesses does your algorithm have (if someone breaks the promise)? Does it always protect the secret of honest participants from dishonest ones?

#### 2.8.2 Solution

Here we give an idea of the solution proposed by Mikhail Kudinov (Bauman Moscow State Technical University).

First of all, it is supposed that no one can buy a quantum computer himself without other participants. Let us assume that $N'$ is the amount of money that one needs to buy a quantum computer and $N = nN'$, where $n$ is the number of participants. The millionaire gave them $X_i$
money for \( i \in \{1, \ldots, n\} \). Each of participants chooses random secrets \( s_{i,j} \) uniformly so that

\[
\sum_{j=1}^{n} s_{i,j} \equiv X_i \pmod{N}.
\]

Then each of then gives the share \( s_{i,j} \) to the owner of \( X_j \) by the secure channel. After this procedure, the owner of \( X_i \) has shares \( s_{k,i} \) for each \( k \in \{1, \ldots, n\} \). It is obvious that

\[
\sum_{j=1}^{n} \sum_{i=1}^{n} s_{i,j} = \sum_{i=1}^{n} X_i \pmod{N}.
\]

Under the first suggestion, all participants can together calculate the common amount of money.

The main disadvantage of the algorithm, in addition to the suggestion, is a big amount of private communication (though the number of keys can be \( n \) for asymmetric schemes).

Analogically, many participants described algorithms similar to Schneier’s calculating average salary algorithm [13]. In general, all such algorithms are vulnerable if \( n - 1 \) participants are dishonest. Some participants tried to describe a possibility to use a cryptosystem, that is homomorphic by “+” and preserves relation “<”, as a general analysis.

The problem of the first school round is the same problem for \( n = 3 \) (score assignment was more loyal). Despite there was a quite big number of solutions for this problem in the student round, each solution had big or small lacks in analysis of the general case, in analysis of the algorithm advantages and disadvantages, in description of communications (number of private communications, what kind of cryptography is used, number of required private keys) and so on. As a result, there was no possibility to chose “best of the best” for 6 scores and we decided to give 5 scores as maximum. There were nine maximal-scored solutions.

### 2.9 Problem “16QAM”

#### 2.9.1 Formulation

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):
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!

2.9.2 Solution

Some details in the problem statement are insignificant. Namely, we could omit the step with the Gray code and mind that Alice substitutes 7-bit codewords of the Hamming code for each symbol in each part of the plaintext.

The crucial idea to break the cipher Alice and Bob use is analyzing the frequency distribution in each part of the ciphertext. This helps to deduce the probable meaning of the most common symbols and form partial words. Tentative search for combinations of consonants and vowels giving actual words in English expands the partial solution. Frequencies of pairs of letters also give an improvement but it could seem inessential. At last, one can employ search engine on the Internet to find the fragment of the book that Alice sent to Bob.

Let us consider a possible solution. Alice uses the Hamming code with the parity check matrix $H$ and the corresponding generator matrix $G$, where

$$ H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}, \quad G = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}. $$

First, rewrite each part of the given ciphertext in the binary form. Split them into 7-bit words and correct errors using the parity check matrix $H$. One can decode the Hamming code into a 4-bit Gray code but it is not a necessary step for the solution. Calculating frequencies of codewords separately in each part of the given ciphertext, we put them in Table 7.
### Table 7: Frequencies of Hamming codewords in the text

| Gray code | Hamming code | Frequency |
|-----------|--------------|-----------|
| 1011      | 0110011      | 46        |
| 0010      | 0101010      | 30        |
| 1001      | 0011001      | 24        |
| 0001      | 1101001      | 24        |
| 0011      | 1000011      | 19        |
| 0000      | 0000000      | 15        |
| 0110      | 1100110      | 13        |
| 1100      | 0111100      | 8         |
| 1111      | 1111111      | 8         |
| 1101      | 1010101      | 7         |
| 0100      | 1001100      | 6         |
| 1110      | 0010110      | 5         |
| 1010      | 1011010      | 5         |
| 0101      | 0100101      | 4         |
| 1000      | 1110000      | 4         |
| 0111      | 0001111      | 2         |

| Gray code | Hamming code | Frequency |
|-----------|--------------|-----------|
| 0100      | 1001100      | 85        |
| 1011      | 0110011      | 50        |
| 1001      | 0011001      | 33        |
| 0001      | 1101001      | 26        |
| 1010      | 1011010      | 17        |
| 0011      | 1000011      | 9         |
| 0000      | 0000000      | 8         |
| 1110      | 0010110      | 7         |
| 1100      | 0111100      | 2         |
| 0010      | 0101010      | 1         |
| 1000      | 1110000      | 1         |
| 0111      | 0001111      | 0         |
| 0101      | 0100101      | 0         |
| 1101      | 1010101      | 0         |
| 0110      | 1100110      | 0         |
| 1111      | 1111111      | 0         |

(a) Part 1  
(b) Part 2

Compare the frequencies obtained with those of letters in the English language. The suitable frequency distribution can be found in [12], which is cited, e. g., at [20]. According to Lewand, arranged from most to least common in appearance, the letters are:

```
et a o i n s h r d l c u m w f g y p b v k j x q z.
```

We start with vowels, punctuation marks, spaces, and a hyphen, which are placed in Part 2. Make a guess that the most frequent symbol in Part 2 is the space. It is also worth to note that most of punctuation marks are followed by a space in contrast to a hyphen, which is usually embraced by letters. Using letter frequencies, we determine the probable spaces, vowels, and hyphen, and construct the following partial solution for this part of the plaintext (the sign # substitutes punctuation):

```
  ee ae e o e u a ia a# e oo oy i o ea ee# u# ea# au ae o i e ea o e aoy a oe
  o i a i eae# a i o o eae a o o i o iee ay ye aeii o aa aie# uuaiy# e uai uy oy
  oe i a e eae ia e eae# i e ee oee e o e a e a e afe# e e a uy ee i a e e o e e a a#
```

Let us turn to Part 1, which contains 16 consonants occurring in the fragment of the book. Let us order the codewords of the Hamming code from most to least frequent in Part 1, as it is shown in Table 7a. Denote the 7-bit codewords by hexadecimal numbers from 0 till F. Then we get the following ciphertext of 220 symbols in length that is splitted into 10 pieces (according to the number sequence given in the task):

```
023402C43E0251412B0103 02C1B32407551003070 4A3 B46 33A4884CE02E84020631094106311739943
1675510A0040C1068047266101D10619FF5FD4031000448090103 355
025108B315023021A3020246102173994 E2333C72410275585D46 021281BD102021A0202631016055
```

Then we match symbol frequencies in Part 1 of the ciphertext with those of consonants in the English alphabet. The first five pairs are like as follows: 0 - t, 1 - n, 2 - s/h, 3 - s/h, 4 - r.

The bigram th is the most frequent in English. This allows us to make a suggestion that 2 substitutes h and 3 substitutes s. Then we obtain a partial solution for Part 1 and, combining with one for Part 2, get the following pieces of the plaintext given in Table 8. It is not difficult to recognize words these are the at the beginning in (1). Also, we can see the as the first word in (2) and (8).
Table 8: Partial plaintext

| No. | Partial plaintext                                      |
|-----|--------------------------------------------------------|
| (1) | thesrthCrsEth5nrnhBtnts                                 |
|     | ee ae e oe o ua iaia#                                  |
| (2) | thCnBshrt755ntts7ts                                   |
|     | e oo oy-oy i o ea ee#                                  |
| (3) | rAs                                                   |
|     | u#                                                    |
| (4) | Br6                                                   |
|     | ea#                                                   |
| (5) | ssAr888rCEthE8trtth6snt9rnt6sn7s99rs                  |
|     | auae o i e ea o e aoy a oe o i a i eae#                |
| (6) | n6755ntAttrtCn6l6nt6nt6n9FF56DrtsnAtttr8t9tnts       |
|     | a i o o o eae a oo o i o iee ay ye aeii o aa aie#      |
| (7) | s55                                                   |
|     | uuay#                                                 |
| (8) | th5nt888sn5thsthnAstthhr6nthn7s99r                   |
|     | e uai uy oy oe i a e ea i e eae#                      |
| (9) | EhssC7hrnth755585Dr6                                  |
|     | i e ee oeee o e a a ee a#                             |
| (10) | thnh8nBnthtnAtth6sntn655                              |
|      | e e a uy ee i a e oe o ee a a#                        |

The best idea for the next step is to search through the English dictionary for words that have given vowels in the prescribed order. It is possible to use one of the tools for pattern recognition available on the Internet, e. g., [19]. Advanced participants of the Olympiad implemented some computer programs on their own.

Consider several examples. We have a word with consonants s55 and vowels uuay in (7), and the last two consonants are identical. The only match is usually, so we assume that 5 substitutes the letter l. The pattern auae in combination with double s gives us two possibilities in (5) — assuage and sausage. In any case, it seems like A means g. Then we have rugs in (3). The pattern uai and consonants 5nt88 lead us to lunatic in (8), so 8 probably means c.

At this point we revise our matching the letters and their frequencies corresponding to the Part 1 of the ciphertext. Let us look at the first eight letters with large frequencies: t n h s r 1 6 7/c. We can see that the letter d has still been hidden. According to the Lewand distribution it is the most probable that 6 means d. Then (4) contains Brd and ea what gives us possible words beard and bread. Therefore, it seems like B substitutes b.

A thoroughly analysis of the remaining ciphertext and search for words by patterns and number of letters eventually lead us to the plaintext (with punctuation replaced by #):

these are the mores of the lunar inhabitants# the moon boy-shorty will not eat sweets# rugs# bread# sausage or ice cream of the factory that does not print ads in newspapers# and will not go to treatment a doctor who did not invented any puzzle advertising to attract patients# usually# the lunatic buys only those things that he read in the newspaper# if he sees somewhere on the wall a clever ad# then he can buy even the thing that he does not need at all#

This is a fragment of the fairytail novel “Dunno on the Moon” by Russian writer Nikolay Nosov. The title character of the novel is a boy-shorty Dunno. The problem was completely solved by 13 teams in the second round and by Samuel Tang (Hong Kong, Black Bauhinia) in the first round.
The best solutions were proposed by the team of Irina Slonkina, Mikhail Sorokin, and Vladimir Bobrov (Bauman Moscow State Technical University), and the team of Vladimir Paprotski, Dmitry Zarembo, and Karina Kruglik (Belarusian State University).

2.10 Problem “APN + Involutions”

The first three questions Q1, Q2, Q3 were given as the problem “APN + Involutions” in the first round. The extended version of the task for the second round included also the question Q4 that contains open problems.

2.10.1 Formulation

Alice wants to construct a block cipher with heavy use of involutions as subcomponents; this minimizes difference between the algorithms for encryption and decryption. She knows that 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 \( n \geq 2 \).

Alice 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 \( i, j \in \{1, \ldots, d\}, i \neq j, 1 \leq d \leq 2^{n-1} \).

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

Q1 Let

\[
\Lambda(g) = \{\alpha_i \oplus \alpha'_i : i = 1, \ldots, d\}, \quad \tilde{\Lambda}(g) = [\alpha_i \oplus \alpha'_i : i = 1, \ldots, d],
\]

\[
B(g) = \{x \oplus y : \{x, y\} \subseteq \text{FixP}(g), x \neq y\}, \quad \tilde{B}(g) = [x \oplus y : \{x, y\} \subseteq \text{FixP}(g), x \neq y],
\]

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

Suppose that \( g \) is an APN permutation. Get necessary conditions for multisets \( \tilde{\Lambda}(g), \tilde{B}(g) \) and sets \( \Lambda(g), B(g) \). Prove that if your conditions do not hold, then \( g \) is not an APN permutation.

Q2 Let

\[
d_{a,b}(g) = |\{x \in \mathbb{F}_2^n : g(x \oplus a) \oplus g(x) = b\}|, \quad a, b \in \mathbb{F}_2^n.
\]

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

Q3 Can you get the nontrivial upper bound on |\text{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 \( n \geq 6 \). 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 $n \geq 3$.

**Remark.** Let us recall 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 $x = (x_1, \ldots, x_n)$, where $x_i \in \mathbb{F}_2$. For two vectors $x, y \in \mathbb{F}_2^n$ their sum is $x \oplus y = (x_1 \oplus y_1, \ldots, x_n \oplus y_n)$, where $\oplus$ stands for XOR operation.
- Let $X = [x_1, \ldots, x_d]$ be a multiset with the underlying set $\mathbb{F}_2^n$, where $x_1, \ldots, x_d \in \mathbb{F}_2^n$. 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 $x \in \mathbb{F}_2^n \setminus \{\alpha, \beta\}$; 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 $s(x + a) \oplus s(x) = b$ has at most 2 solutions.

### 2.10.2 Solution

**Q1** Let $a \in \Lambda(g)$. Hence, $a = x \oplus y$, where $y = g(x)$ and $(x, y) = (\alpha_i, \alpha'_i)$ for some $i$. Then

$$g(x \oplus a) = g(y) = x = y \oplus a = g(x) \oplus a.$$ 

Let $a \in B(g)$. Hence, $a = x \oplus y$, where $x, y \in \text{FixP}(g)$. Then

$$g(x \oplus a) = g(y) = y = x \oplus a = g(x) \oplus a.$$ 

Thus, $d_{x,a}(g) \geq 2$ for any vector $a \in \Lambda(g) \cup B(g)$.

Let $g$ be an APN permutation. Then $d_{x,a}(g) = 2$. Hence, the multiplicity of all elements from $\Lambda(g)$ and $B(g)$ is 1. Thus, $\Lambda(g) = \hat{\Lambda}(g)$ and $B(g) = \hat{B}(g)$. Note that $\Lambda(g) \cap B(g) = \emptyset$.

**Q2** Since $g$ is an APN permutation, then $d_{x,a}(g) \leq 2$. As we get in Q1, $d_{x,a}(g) = 2$ for any vector $a \in \Lambda(g) \cup B(g)$. Let us prove that $d_{x,a}(g) = 0$ for $a \notin \Lambda(g) \cup B(g)$.

Let $a$ be a nonzero vector and $x$ be a solution of $g(x \oplus a) \oplus g(x) = a$. Since $g$ is a permutation, then either $x \in \text{FixP}(g)$ or $x = \alpha_i \ (x = \alpha'_i)$ for some $i$. Consider two cases:

1. Let $x \in \text{FixP}(g)$. Then, $g(x \oplus a) \oplus g(x) = a$ implies $g(x \oplus a) = x \oplus a$. Hence, $x \oplus a \in \text{FixP}(g)$. As a result, $a \in B(g)$.

2. Without loss of generality, let $x = \alpha_i$ for some $i$ and $y = x \oplus a$. If $y \in \text{FixP}(g)$, then $g(x \oplus a) \oplus g(x) = a$ implies $g(x) = x$, which is a contradiction. Hence, without loss of generality, $y = \alpha'_j$ for some $j$ (so, we have $\alpha_i \oplus \alpha'_j = a$). Then

$$g(\alpha_i \oplus a) \oplus g(\alpha_i) = a \Rightarrow g(\alpha'_j) \oplus \alpha'_i = a \Rightarrow \alpha_j \oplus \alpha'_i = a.$$ 

Let us show that $\alpha'_i$ and $\alpha_j$ is also solutions. Indeed,

$$g(\alpha'_i \oplus a) \oplus g(\alpha'_i) = g(\alpha_j) \oplus \alpha_i = \alpha'_j \oplus \alpha_i = a$$ 

and

$$g(\alpha_j \oplus a) \oplus g(\alpha_j) = g(\alpha'_i) \oplus \alpha'_j = \alpha_i \oplus \alpha'_j = a.$$ 

Thus, if $i \neq j$, we get at least 3 solutions that is contradiction for the APN property of $g$. Hence, $j = i$ and $a \in \Lambda(g)$. 

15
Q3 Let us prove that $|\text{FixP}(g)| \leq 1 + (2^{n-1} - 1)^{1/2}$.

The involution $g$ is APN. From Q1 we have

$$B(g) \cap \Lambda(g) = \emptyset.$$  \hspace{1cm} (1)

Let $q = |\text{FixP}(g)|$. Since $g$ is an involution, we have that $q$ is even. From equality (1) and $\Lambda(g) \cup B(g) \subseteq \mathbb{F}_2^n \setminus \{0\}$ it follows that

$$|\Lambda(g)| + |BB(g)| \leq 2^n - 1.$$  \hspace{1cm} (2)

Since $|B(g)| = \binom{n}{2}$, $|\Lambda(g)| = 2^{n-1} - q/2$, we have

$$|\Lambda(g)| + |BB(g)| = q(q - 1)/2 + 2^{n-1} - q/2.$$

From inequality (2), we get

$$q(q - 1)/2 + 2^n - q \leq 2^n - 1.$$

Thus,

$$q(q - 2)/2 \leq 2^{n-1} - 1,$$

i.e.

$$q \leq 1 + (2^{n-1} - 1)^{1/2}.$$  

Q4 (a) It could be computationally verified that $M_2 = \emptyset$ and $|M_3| = 224$. Then, it is known [3] that there are no APN permutations for $n = 4$. Hence, $M_4 = \emptyset$.

(b) Let us recall several definitions. A function $A : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ is affine if $A(x \oplus y) = A(x) \oplus A(y) \oplus A(0)$ for any $x, y \in \mathbb{F}_2^n$. Two functions $F, G : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ are called affine equivalent if there exist affine permutations $A_1, A_2$ such that $F = A_1 \circ F \circ A_2$. It is easy to see that the APN permutation property of a function is an invariant under the affine equivalence. There exist [3] only five the affine equivalence classes of APN permutations. Moreover, by [3, theorem 3] only one class contains functions together with their inverses. Hence, only this class of APN permutations can contain involutions. The representative of this class is the famous inverse function over the finite field: $F(x) = x^{-1}$ for nonzero $x$ and $F(0) = 0$ (here, functions from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$ are considered as functions over the finite field of order $2^n$). The inverse function is an involution. Thus, all APN involutions for $n = 5$ are affine equivalent to the inverse function.

(c) There were no interesting suggestions by the participants for these open problems.

The unique full correct solution in the first round was proposed by Henning Seidler (Germany, TU Berlin). In the second round, the best solution for 11 scores was proposed by the team of Kristina Geut, Sergey Titov, and Dmitry Ananichev (Russia, Ural State University of Railway Transport, Ural Federal University).

2.11 Problem “Sharing”

2.11.1 Formulation

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. Now he is thinking about the following mathematical problem in this approach.
Let $F$ denote the set of invertible functions (permutations) from $\mathbb{F}_2^4$ to $\mathbb{F}_2^4$ and $F^n$ denote the set of invertible functions from $(\mathbb{F}_2^4)^n$ to $(\mathbb{F}_2^4)^n$. Let $F \in F^n$ 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 $f \in F$, a function $F \in F^n$ 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 non-complete sharing exists. Find answers to the following questions!

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

Let $f, g$ be two functions in the same affine equivalence class of $F$ 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 [4] that this equivalence relation partitions $F$ into 302 classes: 1 class corresponds to $A$, 6 classes contain quadratic functions, 295 classes contain cubic functions.

Also, Bob knows that when $n \geq 5$, there exists a non-complete sharing for each $f \in F$ (it can be shown by construction). When $n = 2$ a non-complete sharing exists only for the functions in $A$. When $n = 3$, non-complete sharings exist for $A$ and also for 5 out of the 6 equivalence classes containing quadratic functions. When $n = 4$, non-complete sharings exist for $A$, 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 $f \in F$ 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 $\mathbb{F}_2^5$, $\mathbb{F}_2^6$.

### 2.11.2 Solution

**Q1** Let $f, g$ be two functions in the same affine equivalence class of $F$, that is $g = b \circ f \circ a$ for some $a, b \in A$, and let $F \in F^n$ be a non-complete sharing of $f$. At first, one can notice that since $f, g$ are invertible, the mappings $a, b$ must be invertible as well. Let us denote

$$a(x) = Ax + a', \quad x \in \mathbb{F}_2^4,$$

$$b(x) = Bx + b', \quad x \in \mathbb{F}_2^4,$$

where $A, B$ are nonsingular binary matrices of order $4 \times 4$ and $a', b' \in \mathbb{F}_2^4$. 
Using components functions \( \{F_i\}_{i=1}^n \) of \( F \), we define the invertible function \( G \in F^n \) with components functions

\[
G_j(x_1, x_2, \ldots, x_n) = \begin{cases} 
BF_1(Ax_1 + a', Ax_2, \ldots, Ax_n) + b', & j = 1, \\
BF_j(Ax_1 + a', Ax_2, \ldots, Ax_n), & j \neq 1,
\end{cases}
\]

where \( j = 1, 2, \ldots, n \).

Then for any \((x_1, x_2, \ldots, x_n) \in (\mathbb{F}_2^n)^n\), it holds

\[
\sum_{j=1}^n G_j(x_1, x_2, \ldots, x_n) = BF_1(Ax_1 + a', Ax_2, \ldots, Ax_n) + b' + \\
+ \sum_{j=2}^n BF_j(Ax_1 + a', Ax_2, \ldots, Ax_n) = B \left( \sum_{j=1}^n F_j(Ax_1 + a', Ax_2, \ldots, Ax_n) \right) + b' = \\
= Bf(Ax_1 + a' + Ax_2 + \ldots + Ax_n) + b' = Bf(A \left( \sum_{i=1}^n x_i \right) + a') + b' = \\
= b \circ f \circ a \left( \sum_{i=1}^n x_i \right) = g \left( \sum_{i=1}^n x_i \right).
\]

Therefore, the function \( G \in F^n \) defined as

\[
G(x_1, x_2, \ldots, x_n) = (G_1(x_1, x_2, \ldots, x_n), G_2(x_1, x_2, \ldots, x_n), \ldots, G_n(x_1, x_2, \ldots, x_n)),
\]

is a sharing of \( g \).

From non-completeness of \( F \) it follows that \( G_j \), which is in fact an affine transformation of \( F_j \), does not depend on \( x_j \). Hence, \( G \) is a non-complete sharing of \( g \).

**Q2-Q3** These open problems were not solved completely during the Olympiad. Nevertheless, one perspective solution was proposed by the team of Victoria Vlasova, Mikhail Polyakov, and Alexey Chilikov (Bauman Moscow State Technical University). They found a sufficient condition for the existence of non-complete sharing for \( n = 3 \). Let us describe it here.

Let \( \text{wt}(y) \) be the Hamming weight of a binary vector \( y \). For \( \sigma \in \mathbb{F}_2 \), we denote

\[
\delta_\sigma(y) = \begin{cases} 
y, & \sigma = 1, \\
0, & \sigma = 0,
\end{cases}
\]

where \( 0 \) is a zero vector of the same dimension as \( y \).

Let \( V \) be a vector space over the field \( K \) and assume that for the invertible function \( f : V \to V \) it holds

\[
\sum_{\sigma \in \mathbb{F}_2^n} (-1)^{\text{wt}(\sigma)} f \left( \sum_{i=1}^n \delta_{\sigma_i}(x_i) \right) = 0, \tag{3}
\]

then there exists a non-complete sharing for \( f \). Further we consider the case \( n = 3 \).

Indeed, for any \((x_1, x_2, x_3) \in V^3\) put

\[
F_1(x_1, x_2, x_3) = f(x_2) - f(x_2 + x_3),
\]

18
\[ F_2(x_1, x_2, x_3) = f(x_3) - f(x_1 + x_3), \]
\[ F_3(x_1, x_2, x_3) = f(x_1) - f(x_1 + x_2). \]

It is clear that every \( F_i : V^3 \to V \) does not depend on \( x_i \), where \( i = 1, 2, 3 \). Consider the expression

\[
\sum_{i=1}^{3} F_i(x_1, x_2, x_3) = f(x_2) - f(x_2 + x_3) + f(x_3) - f(x_3 + x_1) + f(x_1) - f(x_1 + x_2) = \sum_{\sigma \in \mathbb{F}_2^3} (-1)^{\text{wt}(\sigma)} f \left( \sum_{i=1}^{3} \delta_{\sigma_i}(x_i) \right) + f(x_1 + x_2 + x_3) - f(0) = f(x_1 + x_2 + x_3) - f(0).
\]

Without loss of generality we assume that \( f(0) = 0 \), otherwise we can consider the initial problem for the function \( g(x) = f(x) - f(0) \) with \( g(0) = 0 \) and which, by the arguments from Q1, has non-complete sharing if and only if \( f \) does.

Finally

\[
\sum_{i=1}^{3} F_i(x_1, x_2, x_3) = f(x_1 + x_2 + x_3),
\]

that concludes the proof.

It was also shown by the authors that the condition (3) is necessary for the existence of non-complete sharing of \( f \) for any \( n \).

Taking \( V = \mathbb{F}_2^m \) with \( m = 4, 5, 6 \) and \( K = \mathbb{F}_2 \) one can obtain a solution of Q2, Q3 for the case \( n = 3 \).

2.12 Problem “Factoring in 2019”

2.12.1 Formulation

Nicole is learning about the RSA cryptosystem. She has chosen random 500-bit prime numbers \( p \) and \( q, 2^{499} \leq p, q < 2^{500} \), 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 \( q \).

\[
n = 40763613025504836845249840044831561583564626405535158138667037
1879167267090530860844304055285019651507728831663677166092475
16155419756121537288444995708421977847213953345126368990185271
10259760189356588305406519080647582874212687596214191915933827
6725209471722241813228925131464750049199632340002019,
\]
2.12.2 Solution

This problem is based on a (simplified) variation of the Coppersmith method. Let \( m = n^2 + 8 \cdot 2019 \). It is a composite number with unknown factors. The idea is to find an integer \( a \) such that numbers
\[
\begin{align*}
  a_1 &= a \cdot 3^{2019} \mod m, \\
  a_2 &= a \cdot 5^{2019} \mod m
\end{align*}
\]
are small enough and \( a_1 p^2 + a_2 q^2 \) exceeds the modulus \( m \) by a small amount and can be recovered from \( a \cdot h \mod m \). This can be done using the Lagrange-Gauss algorithm (which is a special case and the building block of the LLL algorithm). Let \( \Lambda \) be the lattice spanned by the two vectors
\[
\begin{align*}
  v_1 &= (1, (5^{2019} \cdot (3^{2019})^{-1} \mod m)), \\
  v_2 &= (0, m).
\end{align*}
\]
Consider an arbitrary vector \( v = (a_1, a_2) \) in this lattice. It is easy to verify that
\[
a_1 p^2 + a_2 q^2 \equiv a_1 \cdot h \cdot (3^{2019})^{-1} \pmod{m}.
\]
The lattice reduction guarantees to find such vector \( v \) with the norm
\[
\|v\| = \sqrt{a_1^2 + a_2^2} \leq 2^{(d-1)/4} (\det \Lambda)^{1/d} = \sqrt{m}/\sqrt{2},
\]
where \( d = 2 \) is the dimension of the lattice. In particular,
\[
|a_1 p^2 + a_2 q^2| \leq n(p^2 + q^2) < n(p + q)^2 < 10n^2,
\]
where the last two inequalities follow from balancedness of the primes (i.e., \( \max(p, q) \leq 2 \min(p, q) \)).

It follows that there exists an integer \( z, |z| < 10 \), such that
\[
a_1 \cdot h \cdot (3^{2019})^{-1} \mod m + zm = a_1 p^2 + a_2 q^2.
\]
As a result, we obtain an equation in \( p^2 \) and \( q^2 \). By replacing \( p = n/q \), we obtain a biquadratic equation in \( q \) which is easy to solve and factor \( n \).
The final solution is:

\[ p = 201900007587815418168112981041447702468182091751945248792088 \\
90921501144547048007953722271285690350264116081579241189587393 \\
202602664199899594021414383, \]

\[ q = 20190000739734941945213398056820939591822657460839955948263937 \\
53631669289175827851666668014167119439386543289850940734885806 \\
826120718179729242641026893. \]

The best solution was proposed by Alexey Zelenetskiy, Mikhail Kudinov, and Denis Nabokov team (Russia, Bauman Moscow State Technical University).

2.13 Problem “TwinPeaks3” (online)

2.13.1 Formulation

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:

\[
\begin{align*}
(a, b, c, d) &\leftarrow (b, c, d, a \oplus (F_i(b, c, d))) \\
F_i &= F_1 \text{ for odd rounds and } F_i = F_2 \text{ for the rest.}
\end{align*}
\]

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

\[ Y = \text{e473f19a247429ab33b66268d57dd241} \]

(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 [17]. 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 \(Y\).

### 2.13.2 Solution

Let \(f_i\) be the round transformation of round \(i\):

\[f_i : (a, b, c, d) \leftarrow (b, c, d, a \oplus (F_k(i)(b, c, d))),\]

where \(k(i) = 1\) for odd \(i\) and \(k(i) = 2\) for the rest.

Hence, we can represent the encryption transformation \(E\) as

\[E = (f_1f_2)^{16}.\]

Let \(I\) be the incomplete decryption transformation described in the problem statement. The encryption and the incomplete decryption processes only differ in key order, so \(I\) can be written in terms of \(f_i\):

\[I = (f_2f_1)^{16}.\]

The decryption transformation \(E^{-1}\) can be represented as

\[E^{-1} = (f_2^{-1}f_1^{-1})^{16},\]

where \(f_i^{-1}\) is the inverse of \(f_i\) and is given by the following transformation:

\[f_i^{-1} : (a, b, c, d) \leftarrow (d \oplus (F_k(i)(a, b, c)), a, b, c)\]

Thus, to apply \(E^{-1}\) to the ciphertext one should be able to compute \(F_1(x, y, z)\) and \(F_2(x, y, z)\) that are secret. To recover these functions a slide attack can be used.

The idea is to find words \(x = (x_1, x_2, x_3, x_4)\) and \(y = (y_1, y_2, y_3, y_4)\) such that \(f_i(x) = y\). If such a pair is found, then \(F_i\) can be found as

\[F_i(x_2, x_3, x_4) = y_4 \oplus x_1.\]

We use the following idea to find a desired pair: if \(Ef_i(x) = E(y)\), then \(f_i(x) = y\). Let us start with \(F_1\). We need a pair of \(x\) and \(y\) such that \(Ef_1(x) = E(y)\). This relation can be written as

\[(f_1f_2)^{16}f_1(x) = (f_1f_2)^{16}(y)\]

\[f_1(f_2f_1)^{16}(x) = (f_1f_2)^{16}(y)\]

\[f_1I(x) = E(y)\]
We come to a conclusion that if \( f_1 I(x) = E(y) \), then \( f_1(x) = y \). The condition \( f_1 I(x) = E(y) \) can be checked by using the definition of \( f_1 \): if \( (I(x))_2 = (E(y))_1 \), \( (I(x))_3 = (E(y))_2 \) and \( (I(x))_4 = (E(y))_3 \), then it is likely that \( f_1 I(x) = E(y) \). The probability of false positives is approximately \( 2^{-96} \) for random \( F_i \) functions. So, it can be considered as negligible. Both \( I(x) \) and \( E(y) \) are available on the encryption oracle for arbitrary \( x \) and \( y \) as the incomplete decryption and the encryption routines respectively.

To find \( F_i(a,b,c) \), let us brute force over \( x \) and \( y \) of the following forms: \( x = (X,a,b,c) \) and \( y = (a,b,c,X') \). According to the birthday paradox, a desired pair can be found in \( 2^{*2^{16}} \) operations average (instead of \( 2^{32} \) if we lock \( X \) or \( X' \) to some constant value).

As soon as we find such a pair \( x \) and \( y \), we can compute \( F_1(a,b,c) \) and apply \( f_1^{-1} \) to the ciphertext and decrypt the last round. Then \( F_2 \) can be found the same way by replacing \( I \) and \( E \) with each other due to the symmetry. By doing this round by round, we decrypt the whole ciphertext and get the desired message (in hexadecimal notation)

```
acherrypieplease
```

The reference implementation of this attack requires \( 2^{22} \) blocks of text to be encrypted and 10 minutes of time average. It is important to use the server’s ability to process multiple blocks of text at a time to minimize the amount of HTTP requests.

Four teams successfully solved the problem using the same method.

### 2.14 Problem “Cur127”

#### 2.14.1 Formulation

Bob is developing the 3OTA infrastructure and has designed a new hash function \( \text{Curl27} \) for it. A distinguishing feature of the infrastructure is the ternary logic: trits from the set \( T = \{0,1,-1\} \) are used instead of bits, ternary strings and words are used instead of binary ones. The \( \text{Curl27} \) hash function is defined below. Its implementation in Java can be found in [18].

Find a collision for \( \text{Curl27} \), that is, different ternary strings \( X \) and \( X' \) such that \( \text{Curl27}(X) = \text{Curl27}(X') \). 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 \( \text{Curl27} \).** The \( \text{Curl27} \) function maps a ternary string \( X \) of arbitrary length to a hash value from \( T^{243} \). When hashing, an auxiliary sponge function \( \text{Curl27-f}: T^{729} \rightarrow T^{729} \) is used. The hashing algorithm:

1. Pad \( X \) with zeros to make its length a multiple of 243. Divide the resulting string into blocks \( X_1,X_2,\ldots,X_d \in T^{243} \).

2. Prepare the state \( W = W_0W_1W_2 \in T^{729} \) consisting of words \( W_i \in T^{243} \). 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 \( \overline{1010000} \ldots 0 \). Here \( \overline{1} \) stands for \(-1 \).

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

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

$$S: \mathbf{T}^3 \to \mathbf{T}^3, \ (a, b, c) \mapsto (F(a, b, c), F(b, c, a), F(c, a, b))$$

is used. Here

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

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 $S(a, b, c)$.

Groupings are organized as follows (see the picture below). 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.

**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 011011...011 is written in each of them instead.

2.14.2 Solution

For a word $u$ in the alphabet $\mathbf{T}$, let $u^m$ be the word of $m$ copies of $u$. Supposing $u = u_0u_1\ldots u_{n-1}$ denote $u^{[m]} = u_0^mu_1^m\ldots u_{n-1}^m$. We call a word of the form $u^{[m]}$ $m$-fragmented.

**Theorem.** Let $m$ be a power of 3, $m \leq 729$. The sponge function Curl27-f preserves $m$-fragmentation, that is, if $W$ is $m$-fragmented, then Curl27-f($W$) is also $m$-fragmented.

**Proof.** At the $i$th step of the Curl27-f round function, the state $W$ is divided into words of length $n = 3^{6-i}, \ i = 1, 2, \ldots, 6$. For $n \leq m$ the step function preserves equality of trits inside fragments. It follows from the fact that $S(a, a, a) = (b, b, b)$. For $n > m$ equality is also preserved since in each fragment trits at the different positions are processed in the same way. \[\square\]
Let $m$ be a small power of 3 (interesting cases are $m = 3, 9, 27$). Consider a ternary string $X$ of length

$$1 + 3 + 3^2 + \ldots + 3^{m-1} = (3^m - 1)/2.$$ 

The length is given by a word of $m$ ones. Consequently, the initial state of Curl27 when processing $X$ is $m$-fragmented (one fragment of ones, the remaining fragments of zeros).

Let us choose trits of $X$ so as to preserve $m$-fragmentation of the state during hashing. This is easy to do using Theorem: each full $m$-fragment of $X$ must have the form $\alpha^m$, $\alpha \in \mathbb{T}$, and, in addition, trits of the last (incomplete) fragment must be zero to be consistent with the padding trits. Having achieved $m$-fragmentation of states, we automatically obtain $m$-fragmentation of hash values. Now a hash value is determined by $243/m$ trits, each of which is repeated $m$ times. We can find a collision for Curl27 after processing of about $\sqrt{3^{243/m}}$ strings $X$ of the described structure, that is, in time of order

$$3^m \cdot \sqrt{3^{243/m}} = 3^{m+121.5/m}.$$ 

The minimum of the function above is achieved at $m = 9$. During the attack with $m = 9$ it is required to process approximately $\sqrt{3^{13.5}}$ strings of 9841 = 243 · 40 + 121 trits each.

An example of colliding messages:

$$X = 0^{243-39}(10110011010111100101100000)[9]0^{121},$$

$$X' = 0^{243-39}(00001111010011111001000000)[9]0^{121}.$$ 

This collision was found by Jeremy Jean (National Cybersecurity Agency of France), the only participant who solved the problem.

The preservation of fragmentation is an invariant of Curl27-f which allows to decrease the dimension and thereby effectively solve the basic problem. To solve the bonus problem, Jeremy Jean proposed to use another invariant for Curl27-f: if each part $W_0$, $W_1$, $W_2$ of the state $W$ is 3-expanded, then this fact also holds for Curl27-f($W$). Here we call a word $U \in \mathbb{T}^{243}$ 3-expanded if it has the form $(abc)^{81}$, $abc \in \mathbb{T}^3$.

In the initial state, the parts $W_0$ and $W_2$ are indeed 3-expanded. To comply with the invariant, the part $W_1$ representing the length of a hashed string $X$ must have one of the forms $(abi)^{81}$, $(a10)^{81}$ or $(100)^{81}$ (the length is nonzero and positive). As a result, $X$ consists of at least $1 + 27 + \ldots + 27^{80} > 3^{240}$ trits.

It is easy to maintain the invariant during hashing: full 243-fragments of $X$ must be 3-expanded and the last incomplete fragment (if it exists) must be filled with zeros. The resulting hash values are 3-expanded, there are only 27 choices for them and a collision will surely be found after processing only 28 strings $X$. Of course, the attack is impractical: the time of order $3^{240}$, which is required only for recording colliding messages, is unacceptably large even compared to the time $3^{243/2}$ of the standard birthday attack.

### 2.15 Problem “8-bit S-box”

#### 2.15.1 Formulation

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 $S$. 

25
2. **The nonlinearity of** $S$, denoted by $\text{nl}(S)$, 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 $\text{du}(S)$ is the maximal number of solutions of the equation $S(x) \oplus S(x \oplus \alpha) = \beta$ for any nonzero vector $\alpha$ and any vector $\beta$.

4. **The (graph) algebraic immunity** of $S$, denoted by $\text{ai}(S)$, 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 $S_{\text{AES}}$:

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

The value $\text{ai}(S_{\text{AES}}) = 2$ means that $S_{\text{AES}}$ (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 $(\text{deg}, \text{ai})(S) = (7, 3)$ (optimal values), we need to maximize $\text{nl}(S)$ and minimize $\text{du}(S)$. The current best result [7, 8] is

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

**Problem for a special prize!** You need to improve this result: find 8-bit $S$ with $\text{nl}(S) > 108$ and/or $\text{du}(S) < 6$ while preserving $\text{deg}(S) = 7$ and $\text{ai}(S) = 3$.

**Remarks.** Let us recall 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: $f(x) = \bigoplus_{I \in \mathcal{P}(N)} a_I \left( \prod_{i \in I} x_i \right)$, 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: $\text{deg}(F) = \max \{|I| : a_I \neq 0, \ I \in \mathcal{P}(N)\}$.

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 $S$.

### 2.15.2 Solution

There were no valuable ideas from the Olympiad participants. The problem remains unsolved for the considered configuration of cryptographic properties. There exist several dozen of constructions, based on well-known butterfly structure, that provide current record $(7, 108, 6, 3)$, see [7, 8]. This leads to the idea that if candidates for improvement exist, then they are likely outside the known structures and constructions of cryptographic permutations.

### 2.16 Problem “Conjecture”

#### 2.16.1 Formulation

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 $\beta$ 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 $\beta$ 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.

### 2.16.2 Solution

This mathematical problem is open and difficult. It was presented in [5] for the first time and discussed in [6]. The conjecture was verified for small $n$ (odd values $n \leq 11$, even values $n \leq 8$). The Olympiad participants suggested several ideas. Unfortunately, none of them gave significant advances to prove a conjecture or search for a counterexample. The team of Kristina Geut, Sergey Titov, and Dmitry Ananichev (Ural State University of Railway Transport) and the team of Alexey Zelenetskiy, Mikhail Kudinov, and Denis Nabokov (Bauman Moscow State Technical University) proved the conjecture for a particular case $k = 1$. Nevertheless, this case is peculiar since the function is then quadratic and the result is known for quadratic functions. The proofs cannot be generalized to the common case.

### 3 Winners of the Olympiad

Here we list information about the winners of NSUCRYPTO’2019 in Tables 9, 10, 11, 12, 13.

**Table 9: Winners of the first round in school section A (“School Student”)**

| Place | Name                | Country, City     | School                                             | Scores |
|-------|---------------------|-------------------|----------------------------------------------------|--------|
| 1     | Borislav Kirilov    | Bulgaria, Sofia   | The First Private Mathematical Gymnasium           | 16     |
| 1     | Alexey Lvov         | Russia, Novosibirsk | Gymnasium 6                                  | 16     |
| 2     | Lenart Bucar        | Slovenia, Ljubljana | Gymnasium Bezigrad                             | 15     |
| 3     | Varvara Lebedinskaya | Russia, Novosibirsk | The Specialized Educational Scientific Center of Novosibirsk State University | 14     |
| 3     | Gabriel Ericson     | Sweden, Örebro    | Tullangsskolan                                   | 14     |
| Diploma | Vlad Conescu      | Romania, Slatina  | Radu Greceanu National College                    | 11     |
| Diploma | Wang Duanyu       | Singapore, Singapore | New Town Primary School                         | 9      |
| Diploma | Vlad Ratnikov     | Russia, Yaroslavl | School 33 of Yaroslavl                           | 9      |
| Diploma | Nikita Kukin       | Russia, Moscow    | Gymnasium 1540 of Moscow                        | 8      |
| Diploma | Michail Kostochka  | Russia, Novosibirsk | Lyceum 130                                     | 8      |
Table 10: **Winners of the first round, section B (in the category “University Student”)**

| Place | Name              | Country, City      | University                                | Scores |
|-------|-------------------|--------------------|-------------------------------------------|--------|
| 1     | Maxim Plushkin    | Russia, Moscow     | Lomonosov Moscow State University         | 22     |
| 1     | Mikhail Kudinov   | Russia, Moscow     | Bauman Moscow State Technical University  | 21     |
| 2     | Narendra Patel    | India, Roorkee     | Indian Institute of Technology Roorkee    | 19     |
| 2     | Vladimir Schavelev| Russia, Saint Petersburg | Saint Petersburg State University | 19     |
| 3     | Thanh Nguyen Van  | Vietnam, Ho Chi Minh City | Ho Chi Minh City University of Technology | 16     |
| 3     | Narendra Patel    | India, Roorkee     | Indian Institute of Technology Roorkee    | 19     |
| 3     | Vladimir Schavelev| Russia, Saint Petersburg | Saint Petersburg State University | 19     |
| 3     | Than Nguyen Van   | Vietnam, Ho Chi Minh City | Ho Chi Minh City University of Technology | 15     |

Diploma

- Denis Nabokov
  - Russia, Moscow
  - Bauman Moscow State Technical University
  - 14
- Filip Dashtevski
  - Macedonia, Kumanovo
  - TU Delft
  - 14
- Sayooj Samuel
  - India, Kollam
  - Amrita University
  - 14
- Paul Cotan
  - Romania, Iași
  - Alexandru Ioan Cuza University
  - 13
- Karina Kruglik
  - Belarus, Minsk
  - Belarusian State University
  - 13
- Hosein Hadipour
  - Iran, Tehran
  - University of Tehran
  - 13
- Polina Raspopova
  - Russia, Yekaterinburg
  - Ural State University of Railway Transport
  - 12
- Gorazd Dimitrov
  - Macedonia, Skopje
  - Ecole Polytechnique
  - 12
- Diana Bespechnaya
  - Russia, Moscow
  - Bauman Moscow State Technical University
  - 12
- Nikolay Prudkovskiy
  - Russia, Moscow
  - Bauman Moscow State Technical University
  - 12
- Riccardo Zanotto
  - Italy, Pisa
  - University of Pisa
  - 12
- Dmitry Zakharov
  - Russia, Moscow
  - National Research Nuclear University MEPhI
  - 12

Table 11: **Winners of the first round, section B (in the category “Professional”)**

| Place | Name              | Country, City      | Organization                          | Scores |
|-------|-------------------|--------------------|---------------------------------------|--------|
| 1     | Henning Seidler   | Germany, Berlin    | TU Berlin                             | 26     |
| 2     | Samuel Tang       | Hong Kong, Hong Kong | Black Bauhinia                    | 20     |
| 2     | Madalina Bolboceanu | Romania, Bucharest | Bitdefender                           | 20     |
| 3     | Irina Slonkina    | Russia, Moscow     | National Research Nuclear University MEPhI | 16     |

Diploma

- Harry Lee
  - Hong Kong, Hong Kong
  - Blocksquare Limited
  - 14
- Alexey Chilikov
  - Russia, Moscow
  - Bauman Moscow State Technical University
  - 14
- Victoria Vlasova
  - Russia, Moscow
  - Bauman Moscow State Technical University
  - 14
- Darko Ninkovic
  - Serbia, Belgrade
  - University of Belgrade
  - 13
- Dheeraj M Pai
  - India, Chennai
  - Hyperweb Media Private Limited
  - 13
- Dmitry Ananichev
  - Russia, Yekaterinburg
  - Ural Federal University
  - 13
- Ekaterina Kulikova
  - Germany, Munich
  - 13
- George Teseleanu
  - Romania, Bucharest
  - Institute of Mathematics of the Romanian Academy
  - 12
Table 12: **Winners of the second round (in the category “University student”)**

| Place | Name                                      | Country, City                | University                                        | Scores |
|-------|-------------------------------------------|------------------------------|---------------------------------------------------|--------|
| 1     | Alexey Zelenetskiy, Mikhail Kudinov, Denis Nabokov | Russia, Moscow              | Bauman Moscow State Technical University          | 51     |
| 2     | Ngoc Ky Nguyen, Dung Truong, Phuoc Nguyen Ho Minh | Vietnam, Ho Chi Minh City; France, Paris | Ho Chi Minh City University of Technology, Ecole Normale Superieure | 43     |
| 2     | Thanh Nguyen Van, Quoc Bao Nguyen, Ngan Nguyen | Vietnam, Ho Chi Minh City    | Ho Chi Minh City University of Technology          | 40     |
| 3     | Maxim Plushkin                            | Russia, Moscow              | Lomonosov Moscow State University                 | 34     |
| 3     | Ilya Trusevich, Maxim Bibik, Alexander Shulga | Belarus, Minsk              | Belarusian State University                       | 38     |
| Diploma | Paul Cotan, Evgnosia-Alexandra Kelesidis | Romania, Iași               | Alexandru Ioan Cuza University                    | 26     |
| Diploma | Roman Sychev, Diana Bespechnaya, Nikolay Prudkovskiy | Russia, Moscow              | Bauman Moscow State Technical University          | 24     |
| Diploma | Vladimir Paprotski, Dmitry Zaremba, Karina Kruglik | Belarus, Minsk              | Belarusian State University                       | 21     |
| Diploma | Vitaliy Cherkashin, Zoya Tabikhanova, Evgenia Bykova | Russia, Novosibirsk        | Novosibirsk State Pedagogical University          | 18     |

Table 13: **Winners of the second round (in the category “Professional”)**

| Place | Names                                      | Country, City                | Organization                                                      | Scores |
|-------|--------------------------------------------|------------------------------|-------------------------------------------------------------------|--------|
| 1     | Irina Slonkina, Mikhail Sorokin, Vladimir Bobrov | Russia, Moscow              | Bauman Moscow State Technical University                          | 48     |
| 1     | Kristina Geut, Sergey Titov, Dmitry Ananchev | Russia, Yekaterinburg        | Ural State University of Railway Transport, Ural Federal University | 46     |
| 2     | Henning Seidler, Katja Stumpp               | Germany, Berlin              | Berlin Technical University                                       | 42     |
| 3     | Victoria Vlasova, Mikhail Polyakov, Alexey Chilikov | Russia, Moscow              | Bauman Moscow State Technical University                          | 37     |
| 3     | Duc Tri Nguyen, Quan Doan, Tuong Nguyen    | Vietnam, Ho Chi Minh City    | Cryptographic Engineering Research Group, pwphofun, Ho Chi Minh City University of Technology | 36     |
| 3     | Madalina Bolboceanu, Andrei Mogage, Radu Titiu | Romania, Bucharest          | Bitdefender, Alexandru Ioan Cuza University                      | 34     |
| Diploma | Elena Kirshanova, Semyon Novoselov, Nikita Kolesnikov | Russia, Kaliningrad         | Immanuel Kant Baltic Federal University                          | 28     |
| Diploma | Vyacheslav Salmanov, Evgeniya Ishchukova, Nikita Kutovoy | Russia, Taganrog           | Southern Federal University                                      | 22     |
| Diploma | Jeremy Jean                                 | France, Paris               | National Cybersecurity Agency of France                          | 20     |
| Diploma | Khai Hanh Tang, Pham Phuong, Yi Tu          | Singapore, Singapore        | Nanyang Technological University                                 | 21     |
| Diploma | Harry Lee, Samuel Tang                       | Hong Kong, Hong Kong        | Black Bauhinia                                                   | 20     |
| Diploma | Danh Nam Tran, Thu Hien Chu Thi, Phu Nghia Nguyen | Vietnam, Ho Chi Minh City | Ho Chi Minh City Pedagogical University, Japan Advanced Institute of Science and Technology, Ho Chi Minh City University of Technology | 20     |
References

[1] Agievich S., Gorodilova A., Idrisova V., Kolomeec N., Shushuev G., Tokareva N. Mathematical problems of the second international student’s Olympiad in cryptography. Cryptologia. 2017, V. 41, No. 6, pp. 534–565.

[2] Agievich S., Gorodilova A., Kolomeec N., Nikova S., Preneel B., Rijmen V., Shushuev G., Tokareva N., Vitkup V. Problems, solutions and experience of the first international student’s Olympiad in cryptography. Prikladnaya Diskretnaya Matematika (Applied Discrete Mathematics). 2015, No. 3, pp. 41–62.

[3] Brinkmann M., Leander G. On the classification of APN functions up to dimension five. Designs, codes and cryptography. 2008, V. 49, pp. 273–288.

[4] De Canni’ere C. “Analysis and Design of Symmetric Encryption Algorithms,” Ph.D. thesis, 2007.

[5] Carlet C. Componentwise APNness, Walsh uniformity of APN functions, and cyclic-additive difference sets. Finite Fields and Their Applications. 2018, V. 53, pp. 226–253.

[6] Carlet C. On APN exponents, characterizations of differentially uniform functions by the Walsh transform, and related cyclic-difference-set-like structures. Proceedings of WCC 2017. Designs, Codes and Cryptography (Postproceedings of WCC 2017). V. 87 (2), pp. 203–224, 2018.

[7] de la Cruz Jiménez R. A. 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, V. 11368, pp. 191–206.

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

[9] Geut K., Kirienko K., Sadkov P., Taskin R., Titov S. On explicit constructions for solving the problem “A secret sharing”. Prikladnaya Diskretnaya Matematika. Prilozhenie. 2017, No. 10, pp. 68–70 (in Russian).

[10] Gorodilova A., Agievich S., Carlet C., Gorkunov E., Idrisova V., Kolomeec N., Kutsenko A., Nikova S., Oblakhov A., Picek S., Preneel B., Rijmen V., Tokareva N. Problems and solutions of the Fourth International Students Olympiad in Cryptography (NSUCRYPTO). Cryptologia. 2019, V. 43, I. 2, pp. 138–174.

[11] Gorodilova A., Agievich S., Carlet C., Hou X., Idrisova V., Kolomeec N., Kutsenko A., Mariot L., Oblakhov A., Picek S., Preneel B., Rosie R., Tokareva N. The Fifth International Students’ Olympiad in Cryptography - NSUCRYPTO: problems and their solutions. Cryptologia. 2020, V. 44, I. 3, pp. 223–256.

[12] Lewand R. E. Cryptological Mathematics, MAA, Washington, 2000.

[13] Schneier B. Applied Cryptography: Protocols, Algorithms and Source Code in C. Wiley; 2nd edition, 1996.

[14] Tokareva N., Gorodilova A., Agievich S., Idrisova V., Kolomeec N., Kutsenko A., Oblakhov A., Shushuev G. Mathematical methods in solutions of the problems from the Third International Students’ Olympiad in Cryptography. Prikladnaya Diskretnaya Matematika (Applied Discrete Mathematics). 2018, No. 40, pp. 34–58.
[15] https://nsucrypto.nsu.ru/

[16] https://nsucrypto.nsu.ru/unsolved-problems/

[17] https://nsucrypto.nsu.ru/archive/2019/round/2/task/4/

[18] https://nsucrypto.nsu.ru/media/Olympiads/2019/Round_2/Tasks/curl27.java

[19] Find Words Using Pattern Matching, in *Litscape.com*. Available at http://www.litscape.com/word_tools/pattern_match.php.

[20] Letter Frequency, in *Wikipedia*. Available at https://en.wikipedia.org/wiki/Letter_frequency.

[21] https://www.ibm.com/blogs/research/2018/01/quantum-prizes/