ВЕСТНИК
КЕМЕРОВСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА
СЕРИЯ: ГУМАНИТАРНЫЕ И ОБЩЕСТВЕННЫЕ НАУКИ

оригинальная статья

Методы кодирования информации на базе трехмерного параллелепипеда на уроках математики

Сергеева Ольга Алексеевна
Кемеровский государственный университет, Россия, Кемерово
https://orcid.org/0000-0002-9204-6318
okoin@yandex.ru

Кутовая Анастасия Сергеевна
Кемеровский государственный университет, Россия, Кемерово

Поступила 04.07.2023. Принята после рецензирования 25.09.2023. Принята в печать 25.09.2023.

Аннотация: Активно развивающийся процесс информатизации общества требует усиления мер по обеспечению информационной безопасности. Организации и частные лица должны своевременно принимать необходимые меры по защите своей информации и минимизации рисков ее потери. Одним из решений этой задачи является включение соответствующего учебного материала по элементам информационной безопасности, в частности по методам кодирования и шифрования информации, в программу школьных курсов математики и информатики. Цель – предложить оригинальный надежный способ кодирования текста защищаемой информации, доступный для изучения в рамках школьного курса математики в 9–11 классах. Для разработки этого способа были использованы геометрические методы представления точек в трехмерном пространстве, ограниченном замкнутым параллелепипедом, и элементарные алгебраические операции с матрицами. В работе предложены поточные и k-граммные варианты матричного способа кодирования текста и их программная реализация на языке программирования Python. Предложенные методы кодирования доступны к освоению учащимися 9–11 классов общеобразовательных школ, так как не требуют специальной математической подготовки. Освоение данных методов в процессе обучения математике и их последующее практическое использование окажется полезным для повышения уровня личной информационной безопасности учащегося и может способствовать формированию у него алгоритмического стиля мышления и развитию математических способностей. Представленные задания для самостоятельных работ учащихся позволят закрепить практические навыки кодирования текста. Результаты статьи могут быть использованы для разработки учебно-методических материалов по математике при изучении методов кодирования и шифрования информации, а также в рамках учебных дисциплин, связанных с информационной безопасностью.

Ключевые слова: кодирование, алфавит, текст, параллелепипед, матрица, защита информация, шифрование, k-грамма

Цитирование: Сергеева О. А., Кутовая А. С. Методы кодирования информации на базе трехмерного параллелепипеда на уроках математики. *Вестник Кемеровского государственного университета. Серия: Гуманитарные и общественные науки*. 2023. Т. 7. № 4. С. 443–456. https://doi.org/10.21603/2542-1840-2023-7-4-443-456

full article

Methods of Encoding Information Based on Three-Dimensional Parallelepiped in Teaching Mathematics

Olga A. Sergeeva
Kemerovo State University, Russia, Kemerovo
https://orcid.org/0000-0002-9204-6318
okoin@yandex.ru

Anastasia S. Kutovaya
Kemerovo State University, Russia, Kemerovo

Received 4 Jul 2023. Accepted after review 25 Sep 2023. Accepted for publication 25 Sep 2023.

Abstract: Social informatization requires strong information security measures that would allow companies and individuals to protect their data. Courses in information security may solve this problem. However, methods of coding and encryption can be incorporated in the school curriculum as part of Mathematics and Computer Science. This article introduces an original reliable method of encoding to be integrated with the high school course of Mathematics. The authors used geometric methods of representing points in three-dimensional space bounded by a closed parallelepiped and elementary algebraic operations with matrices. The paper proposes flow and k-gram variants of the matrix method of text encoding and their program implementation in the Python programming language. The encoding methods require no special mathematical training, which makes them teachable to 9–11-graders.

By mastering these methods, high-school students can increase their personal information security and develop an algorithmic style of thinking, as well as mathematical skills in general. The article contains some extramural tasks that consolidate practical skills of text coding. The academic and methodological materials make it possible to give the methods of coding and encryption while teaching Mathematics, as well as part of academic disciplines related to information security.
Keywords: encoding, alphabet, text, parallelepiped, matrix, data protection, encryption, k-grams

Citation: Sergeeva O. A., Kutovaya A. S. Methods of Encoding Information Based on Three-Dimensional Parallelepiped in Teaching Mathematics. *Vestnik Kemerovskogo gosudarstvennogo universiteta. Seriia: Gumanitarnye i obshchestvennye nauki*, 2023, 7(4): 443–456. (In Russ.) https://doi.org/10.21603/2542-1840-2023-7-4-443-456

Введение

Во всех сферах деятельности современного человека значительно увеличилось количество обрабатываемой, передаваемой и хранимой информации, поэтому ее защита от несанкционированного доступа стала крайне важной задачей. В связи с этим актуально еще со школьного этапа обучения ориентировать учащихся соблюдать основные принципы преобразования и защиты информации и уметь применять соответствующие меры по их организации [1].

Кодирование текстовой информации – это процесс замены символов текста для облегчения передачи, преобразования и хранения данных. Кодирование текста имеет широкое практическое применение: при передаче и хранении личной информации и конфиденциальных данных, в банковской сфере, при работе с программными приложениями и мессенджерами. Кроме того, кодирование является обязательным этапом, предшествующим шифрованию информации с целью ее защиты от незаконного использования [2].

Для проведения кодирования и последующего шифрования информации ее открытый текст записывается в некотором алфавите, представляющем собой конечное множество знаков, достаточное для написания текста информации. Кодирование позволяет перевести элементы открытого текста в математические объекты (числа, точки, векторы и т. п.), с которыми можно производить математические преобразования.

Кодировать алфавит можно различными способами. Для этого принято использовать:

- порядковые номера символов в используемом алфавите, начиная с нуля или с 10, 100 и т. п. (шифр Цезаря [3], аффинный шифр [4], шифр RSA [4], шифр Хилла [5]);
- координаты символа в алфавитной таблице (шифр Полибия [6]);
- двоичные значения одинаковой размерности (шифр Виженера [7], шифр Бэкона [8]);
- точки кривой (эллиптические шифры [9; 10]).

В современной криптографии появляются новые направления, использующие последние достижения фундаментальных наук, в первую очередь математики и информатики [11]. Однако при этом с активным развитием новых информационных технологий криптографические методы становятся все уязвимее и недостаточно стойкими к всевозможным атакам [12]. Одним из решений этой проблемы является комбинирование разных способов кодирования и шифрования защищаемой информации [13]. В частности, могут быть использованы нестандартные способы кодирования.

Изучение и применение математических способов кодирования и преобразования информации на школьных уроках математики позволяет подготавливать учащихся как грамотных представителей современного информационного общества [14].

В данной работе в рамках школьного курса математики в форме элективного курса или на внеурочных дополнительных занятиях по математике изучен оригинальный матричный способ кодирования текстовой информации. Он основан на определении координат точек в трехмерном пространстве, взаимно-однозначно соответствующем символам алфавита, и составлении квадратных матриц по этим координатам. Каждый символ выбранного алфавита в этом случае будет представлен в виде матрицы размером 2×2. Этот матричный способ кодирования алфавита позволяет кодировать и далее при необходимости шифровать открытые тексты посимвольно или k-граммно. В отличие от классического матричного шифра Хилла, использование матриц непосредственно для кодирования алфавита избавляет от необходимости выполнять предварительные преобразования текста на самом этапе шифрования.

Предложенный здесь способ кодирования текста использует элементарный математический аппарат и с необходимыми пояснениями из алгебры [15–17] доступен к освоению обучающимися общеобразовательных школ в 9–11 классах.

Рассмотренные комбинированные способы кодирования информации и методы их графической визуализации могут быть использованы как учебные материалы для углубленного изучения математики на дополнительных уроках или элективных курсах по математике в общеобразовательных учреждениях [18], а также в учреждениях дополнительного образования. Изучение методов кодирования и шифрования информации на уроках математики и применение

их на практике будет способствовать формированию профессионального самоопределения у учащихся в процессе обучения [19], повышению уровня их личной информационной безопасности и развитию у них пространственного мышления, математических способностей и нестандартного креативного подхода к решению актуальных задач.

В профильных классах углубленное обучение происходит сразу по комплексу предметов [19]. В классах с углубленным изучением математики и информатики можно четко обозначить межпредметную связь и преемственность в развитии общих компетенций и универсальных учебных действий обучающихся. Например, при изучении методов шифрования в курсе математики можно добавить занятия по алгоритмизации и заниматься программной реализацией изученных способов кодирования и шифрования. Таким образом, в процессе обучения будет наглядно проявляться межпредметная связь математики, информатики и криптографии.

Для программной реализации в работе был выбран высокоуровневый интерпретируемый язык программирования Python версии 3.10 [20]. Данный язык программирования прост и гибок в применении, его синтаксис отличается от других языков программирования своей лаконичностью и наглядностью. Вложенность операторов обозначается отступами, что повышает читаемость кода и избавляет от использования лишних операторных скобок. Все это способствует быстрому освоению языка Python, благодаря чему его чаще других языков программирования рекомендуют для программной реализации своих учебных или научных проектов непрофессиональным начинающим программистам, в том числе школьникам [21].

При написании программного кода дополнительно была использована свободная библиотека NumPy версии 1.21.6, которая к возможностям встроенного Python добавляет поддержку многомерных массивов большого размера, а также матриц и математических функций для операций с массивами [22].

В материалах работы также приводится программная реализация описанных способов кодирования на языке программирования Python и предложены соответствующие задания для самостоятельных работ учащихся.

Результаты

В статье рассмотрены два варианта матричного способа кодирования открытого текста: на базе параллелепипеда и на базе куба, а также вариант блочного кодирования текста с использованием *k*-грамм. Ниже приводятся описания каждого варианта кодирования в трехмерном пространстве, методы их графической визуализации, соответствующие примеры, программный код на языке программирования Python и учебно-методические материалы.

Алфавит на базе параллелепипеда

Для матричного кодирования открытого текста каждому символу алфавита поставим в соответствие точку из трехмерного пространства с целочисленными координатами (x, y, z), где $x, y, z \in N$. Однако точек с такими координатами во всем пространстве R^3 бесконечно много, поэтому их множество необходимо ограничить до конечного. Для этого можно использовать ограниченный трехмерный параллелепипед, все точки которого на границе и внутри, удовлетворяющие условию $x, y, z \in N$, будут использоваться для кодирования символов алфавита (рис. 1).

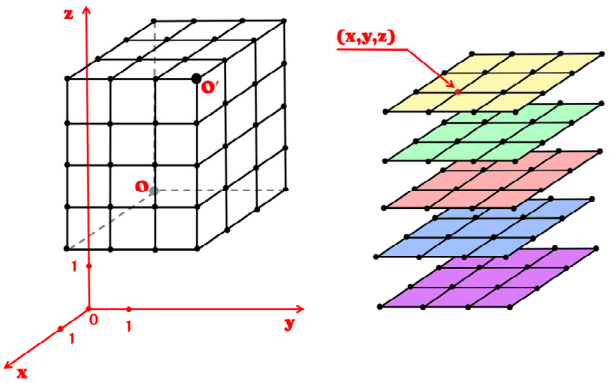

Рис. 1. Трехмерный параллелепипед
Fig. 1. Three-dimensional parallelepiped

Параллелепипед в пространстве можно построить, если задать две его диагонально противоположных точки: нижнюю левую точку и верхнюю правую точку. Назовем их начальной и конечной точками соответственно и обозначим $O = (x_0,y_0,z_0)$ и $O' = (x',y',z')$. Эти точки являются входными параметрами системы и заранее выбираются участниками кодирования.

При этом для корректного кодирования необходимо потребовать, чтобы количество точек с целочисленными координатами в параллелепипеде (внутри и на границе) было равно мощности M используемого алфавита, т. е. равно количеству символов в алфавите. Это означает, что произведение разностей между координатами конечной и начальной точек, увеличенных на единицу, должно быть равным M:

$$(x' - x_0 + 1)\,(y' - y_0 + 1)\,(z' - z_0 + 1) = M. \qquad (1)$$

Для координат точки (x, y, z), соответствующей данному символу алфавита, составим квадратную матрицу 2×2 по следующему правилу:

$$\begin{pmatrix} x & y \\ z & x \end{pmatrix}, \qquad (2)$$

BULLETIN
KEMEROVO STATE UNIVERSITY
HUMANITIES AND SOCIAL SCIENCES

где по главной диагонали матрицы расположены элементы a_{11}, a_{22}, равные первой координате x точки, на вспомогательной диагонали на месте элемента a_{12} – вторая координата y, на месте элемента a_{21} – третья координата z.

Визуально представить алфавит в трехмерном пространстве достаточно трудно, поэтому для наглядности разделим построенный параллелепипед на составляющие его горизонтальные плоскости, как показано на рисунке 1. По свойству параллелепипеда границу и внутреннюю часть плоскостей дискретно заполняют точки с целочисленными координатами. Проведя координатные линии через эти точки в каждой плоскости, получим графические таблицы, которые можно использовать для кодирования по аналогии с принципом квадрата Полибия. Для этого в каждую ячейку таблицы вписывается символ, номер столбца в таблице будет показывать первую координату точки (x), соответствующей данному символу, номер строки – вторую координату (y), а номер самой таблицы – третью координату (z). Рассмотрим такой метод визуализации геометрического представления алфавита на следующем примере.

Пример 1. Найти кодированное представление буквы «Щ», если заданы:

1) начальная точка $O = (16, 23, 10)$,
2) конечная точка $O' = (19, 27, 12)$,
3) алфавит состоит из:
   - заглавных букв русского алфавита;
   - специальных символов «_», «.», «,», «!», «?», «:», «;», «-», «+», «=», «*», «(», «)», «\», «/», «>», «<»;
   - цифр: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Решение. Находим разности соответствующих координат:

$x' - x_0 + 1 = 19 - 16 + 1 = 4$ – количество столбцов в таблице,

$y' - y_0 + 1 = 23 - 27 + 1 = 5$ – количество строк в таблице,

$z' - z_0 + 1 = 10 - 12 + 1 = 4$ – количество таблиц.

Строим таблицы и заполняем их символами заданного алфавит (рис. 2).

Находим, где располагается буква «Щ», и записываем ее координаты (18, 24, 11). Затем составляем матрицу, на главной диагонали которой записываются значения координаты x, а на второстепенной диагонали – значения координат y и z, получаем:

$$C_{щ} = \begin{pmatrix} 18 & 24 \\ 11 & 18 \end{pmatrix}.$$

В итоге кодированное представление буквы «Щ» найдено.

Алфавит на базе куба

Частным случаем параллелепипеда является куб (рис. 3), поэтому все, что выполнялось для параллелепипеда, будет выполняться и в случае куба, но при этом появляются дополнительные особенности, позволяющие рассмотреть еще один вариант кодирования.

Сначала определимся, что нужно знать для построения куба в пространстве. Есть два варианта:

1) пользователи выбирают начальную точку $O = (x_0,y_0,z_0)$ и длину a стороны куба;
2) пользователи выбирают константу c и длину a стороны куба.

Как и в случае с параллелепипедом, используя секущие плоскости $z = k$, $k \in N$, разделим куб на составляющие его горизонтальные решетки (рис. 3). По построению куба в узлах решетки находятся точки с целочисленными координатами. Проведя координатные линии через узлы решетки, получим кодировочные таблицы. Для кодирования алфавита в каждую ячейку полученных плоских таблиц вписывается ровно один символ алфавита. Координаты точки в трехмерном пространстве, соответствующей вписанному в ячейку символу, определяются по правилу: номер столбца в таблице указывает первую координату точки (x), номер строки – вторую координату (y), а номер самой кодировочной таблицы дает третью координату (z).

	16	17	18	19
23	А	Б	В	Г
24	Д	Е	Ё	Ж
25	З	И	Й	К
26	Л	М	Н	О
27	П	Р	С	Т

10

	16	17	18	19
23	У	Ф	Х	Ц
24	Ч	Ш	Щ	Ъ
25	Ы	Ь	Э	Ю
26	Я	–	.	,
27	!	?	:	;

11

	16	17	18	19
23	–	+	=	*
24	(	)	\	/
25	>	<	0	1
26	2	3	4	5
27	6	7	8	9

12

Рис. 2. Местонахождение символа «Щ» в алфавите
Fig. 2. Character [Щ] in the alphabet

 https://doi.org/10.21603/2542-1840-2023-7-4-443-456

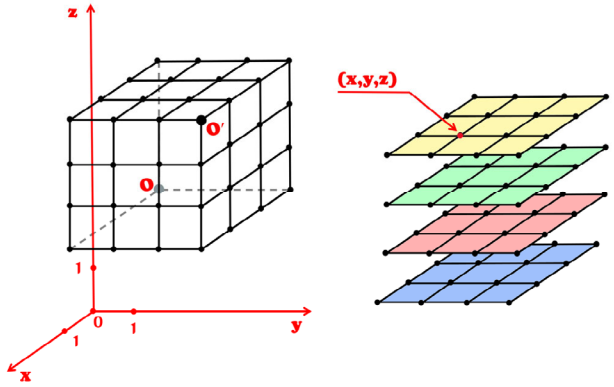

Рис. 3. Трехмерный куб
Fig. 3. Three-dimensional cube

В первом случае надо учитывать, что длина a зависит от количества символов M в выбранном алфавите. Другими словами, а в кубе должна равняться мощности M алфавита

$$a^3 = M. \quad (3)$$

Для демонстрации этого рассмотрим два примера.

Пример 2. Найти матричный код буквы «S», если заданы:

1) начальная точка куба O = (12, 5, 60),
2) длина стороны куба a = 3,
3) алфавит из 27 символов (заглавные буквы английского алфавита и пробел «_»).

Решение. Так как длина стороны куба равна трем, то таблиц также будет три. В каждой таблице – по три столбца и три строки (рис. 4).

Находим, где располагается буква «S», и записываем ее координаты (12, 5, 62). Составляем матрицу по формуле (1):

$$C_S = \begin{pmatrix} 12 & 5 \\ 62 & 12 \end{pmatrix}.$$

Матричный код буквы «S» найден.

Во втором случае константа c будет формировать начальную точку следующим образом: $O = (x_0,y_0,z_0) = (c,c,c)$, а также выполняется условие $a^3 = M$.

Пример 3. Найти кодированное представление символа «%», если заданы:

1) константа c = 41,
2) длина стороны куба a = 4,
3) алфавит состоит из:
   - заглавных букв русского алфавита;
   - специальных символов «_», «.», «,», «!», «?», «:», «;», «>», «<», «(», «)», «-», «+», «=», «%» «*», «”» «\», «/», «@», «№»;
   - цифр: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Решение. Начальная точка имеет координаты O = (41, 41, 41). Так как длина стороны куба равна четырем, то таблиц также будет четыре. В каждой таблице – по четыре столбца и четыре строки (рис. 5).

Находим, где располагается символ «%», и записываем его координаты (44, 44, 43). Составляем матрицу:

$$C_{\%} = \begin{pmatrix} 44 & 44 \\ 43 & 44 \end{pmatrix}.$$

Кодированное представление символа «%» найдено.

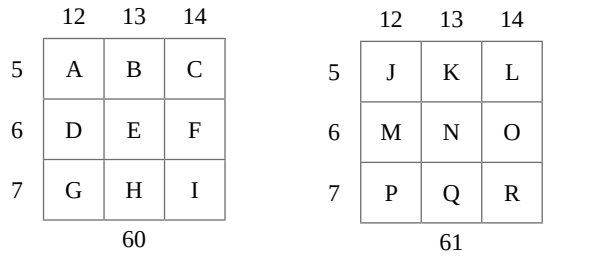

Рис. 4. Местонахождение символа «S» в алфавите
Fig. 4. Character [S] in the alphabet

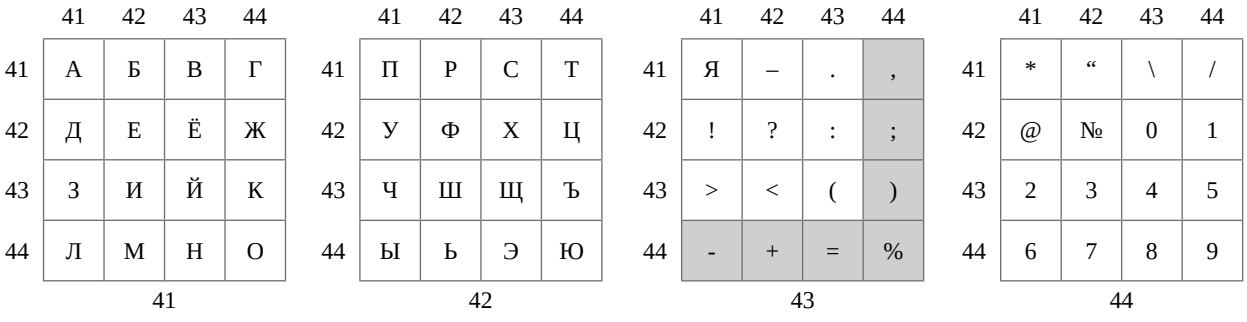

Рис. 5. Местонахождение символа «%» в алфавите
Fig. 5. Character [%] in the alphabet

BULLETIN
KEMEROVO STATE UNIVERSITY
HUMANITIES AND SOCIAL SCIENCES

Матричные *k*-граммы

Для символов алфавита, кодированного на базе трехмерного параллелепипеда (куба), есть возможность использовать матричные *k*-граммы. Формула построения матричной *k*-граммы схожа с формулой числовой *k*-граммы. Но есть один важный нюанс, который нужно учитывать, работая с матричными *k*-граммами и впоследствии используя их для шифрования.

Напомним, что числовой эквивалент k-граммы определяется по формуле:

$$n_j = p^k \cdot x_1 + p^{(k-1)} \cdot x_2 + \ldots + p \cdot x_{(k-1)} + x_k, \quad (4)$$

где $(x_1,x_2,\ldots,x_k)$ – *k*-грамма, x_i – числовой эквивалент *i*-го символа *k*-граммы, $i = 1, \ldots, k$, p – количество символов в используемом алфавите, $j = 1, \ldots, m$; m – количество *k*-грамм в тексте.

В случае построения алфавита на базе параллелепипеда формально формула матричной *k*-граммы не отличается от (4), но число p имеет другой смысл и определяется формулой:

$$p = \max\{x',y',z'\} + 1, \quad (5)$$

где $O' = (x',y',z')$ – координаты конечной точки.

Это связано с тем, что в случае числовых *k*-грамм каждый символ имеет свой единственный порядковый номер, а для матричных *k*-грамм у символа в соответствии будут три числа – целочисленные координаты соответствующей ему точки в трехмерном пространстве.

При работе с кубом для первого случая построения куба (при выбранных начальной точки $O = (x_0,y_0,z_0)$ и длине a значение p будет также вычисляться по формуле (5), где $x' = x_0 + a - 1$, $y' = y_0 + a - 1$, $z' = z_0 + a - 1$.

А во втором случае построения куба (при выбранных константе c и длине a) формула для p принимает вид:

$$p = c + a - 1. \quad (6)$$

Пример 4. Используя кодированный алфавит из примера 1, найдите матричное представление биграммы «БУ» и для проверки выполните обратные действия, чтобы вернуться к исходным символам.

Решение. Найдем матричные представления символов «Б» и «У»:

$$C_Б = \begin{pmatrix} 17 & 23 \\ 10 & 17 \end{pmatrix}, \; C_У = \begin{pmatrix} 16 & 23 \\ 11 & 16 \end{pmatrix}.$$

Найдем $p = \max\{x',y',z'\} + 1 = \max\{19,27,12\} + 1 = 27 + 1 = 28$. Вычисляем матричное представление биграммы «БУ», для этого используем операции умножения матрицы на число и сложения матриц [15]:

$$N_{БУ} = 28 \begin{pmatrix} 17 & 23 \\ 10 & 17 \end{pmatrix} + \begin{pmatrix} 16 & 23 \\ 11 & 16 \end{pmatrix} = \begin{pmatrix} 492 & 667 \\ 291 & 492 \end{pmatrix}.$$

Чтобы получить исходные символы «Б» и «У», нужно поделить матрицу $N_{БУ}$ на $p = 28$. Матрица, состоящая из целых частей элементов $div\left(\frac{1}{28} \cdot N_{БУ}\right)$, будет шифрующей матрицей для первого символа биграммы, где *div* – это целая часть от деления на p. Матрица, состоящая из остатков от деления элементов $mod\left(\frac{1}{28} \cdot N_{БУ}\right)$, будет шифрующей матрицей для второго символа биграммы, где *mod* – это остаток от деления на p. Получаем:

$$C_Б = div\left(\frac{1}{28} \cdot N_{БУ}\right) = div\left(\frac{1}{28} \cdot \begin{pmatrix} 492 & 667 \\ 291 & 492 \end{pmatrix}\right) = \begin{pmatrix} 17 & 23 \\ 10 & 17 \end{pmatrix},$$

$$C_У = mod\left(\frac{1}{28} \cdot N_{БУ}\right) = mod\left(\frac{1}{28} \cdot \begin{pmatrix} 492 & 667 \\ 291 & 492 \end{pmatrix}\right) = \begin{pmatrix} 16 & 23 \\ 11 & 16 \end{pmatrix}.$$

После декодирования исходные символы будут восстановлены.

Программная реализация: матричное кодирование / декодирование текста

Данный программный код можно разделить на две части. Первая часть отвечает за кодирование исходного открытого текста, вторая часть – за декодирование матриц. Рассмотрим каждую в отдельности.

Для начала работы подключаем библиотеку NumPy и все ее инструменты (листинг 1).

Листинг 1. Подключение библиотеки NumPy
Listing 1. Connecting the NumPy Library

```
import numpy as np
```

Для **кодирования** открытого текста используются четыре функции:

1) первая функция *funk1* задает список символов алфавита;
2) вторая функция *funk2* создает словарь, в котором каждому символу алфавита в соответствие ставится точка в пространстве;
3) третья функция *funk3* составляет матрицу, которая кодирует заданный символ алфавита;
4) четвертая функция *funk4* формирует список матриц, в котором каждая матрица соответствует одному символу открытого текста.

Разберем каждую функцию подробнее.

Функция *funk1(alf)* в качестве параметра *alf* принимает строку, которая отвечает за исходный алфавит. Используя цикл *for*, получаем список символов алфавита (листинг 2).

Результатом выполнения этой функции является список символов алфавита (рис. 6).

 https://doi.org/10.21603/2542-1840-2023-7-4-443-456

```
def funk1(alf):
    return [a for a in alf]
```

Листинг 2. Функция *funk1*
Listing 2. *Funk1* function

```
>>> funk1('АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЬЫЪЭЮЯ .,!?:;-+=*()\/><0123456789')
['А', 'Б', 'В', 'Г', 'Д', 'Е', 'Ё', 'Ж', 'З', 'И', 'Й', 'К', 'Л', 'М', 'Н', 'О', 'П', '
Р', 'С', 'Т', 'У', 'Ф', 'Х', 'Ц', 'Ч', 'Ш', 'Щ', 'Ь', 'Ы', 'Ъ', 'Э', 'Ю', 'Я', ' ', '.'
, ',', '!', '?', ':', ';', '-', '+', '=', '*', '(', ')', '\\', '/', '>', '<', '0', '1',
'2', '3', '4', '5', '6', '7', '8', '9']
```

Рис. 6. Результат выполнения функции *funk1*
Fig. 6. Executing the *funk1* function

Следующая функция *funk2(alf, x0, y0, z0, xx, yy, zz)* в качестве параметров принимает последовательность символов в исходном алфавите, координаты начальной точки *O* (второй, третий и четвертый параметры) и конечной точки *O′* (пятый, шестой и седьмой параметры). Внутри функции сначала вызывается функция *funk1*, которая формирует список алфавита. Далее создаются три списка, содержащие числа в промежутке между координатами начальной и конечной точек. Формируется список координат точек параллелепипеда в пространстве, т. е. список списков. Объявляется пустой словарь, ключами которого будут являться символы алфавита, а значениями – координаты точек в пространстве. И через цикл *for* словарь заполняется парами значений *символ – точка* (листинг 3).

В результате выполнения этой функции получаем словарь, в котором каждому символу алфавита в соответствие ставится точка в пространстве (рис. 7).

Функция *funk3(BUKVA, alf, x0, y0, z0, xx, yy, zz)* имеет параметр *BUKVA* строкового типа, а также параметры, отвечающие за алфавит и начальную и конечную точки. С помощью функции *funk2* формируется словарь с парами значений *символ – точка*. Далее находятся координаты заданной буквы в пространстве. Другими словами, переменной *coord* присваивается значение словаря с ключом, равным значению параметра *BUKVA*. Далее при использовании библиотеки NumPy составляется матрица, у которой по диагонали элементы принимают значения первой координаты точки, соответствующей данному ключу, а по вспомогательной диагонали – вторую и третью координаты этой же точки (листинг 4).

В результате выполнения этой функции получаем матрицу, кодирующую заданный символ алфавита (рис. 8).

Рассмотрим функцию *funk4(TEXT, alf, x0, y0, z0, xx, yy, zz)*, у которой добавляется параметр *TEXT*

```
def funk2(alf, x0, y0, z0, xx, yy, zz):
    alf = funk1(alf)
    A = [x for x in range(x0,xx+1)]
    B = [y for y in range(y0,yy+1)]
    C = [z for z in range(z0,zz+1)]
    XYZ = [[x,y,z] for z in C for y in B for x in A]
    ABC = {}
    i = 0
    for a in alf:
        ABC[a] = XYZ[i]
        i += 1
    return ABC
```

Листинг 3. Функция *funk2*
Listing 3. *Funk2* function

```
>>> funk2('АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЬЫЪЭЮЯ .,!?:;-+=*()\/><0123456789',16,23,10,19,27,12)
{'А': [16, 23, 10], 'Б': [17, 23, 10], 'В': [18, 23, 10], 'Г': [19, 23, 10], 'Д': [16, 24, 10
], 'Е': [17, 24, 10], 'Ё': [18, 24, 10], 'Ж': [19, 24, 10], 'З': [16, 25, 10], 'И': [17, 25,
10], 'Й': [18, 25, 10], 'К': [19, 25, 10], 'Л': [16, 26, 10], 'М': [17, 26, 10], 'Н': [18, 26
, 10], 'О': [19, 26, 10], 'П': [16, 27, 10], 'Р': [17, 27, 10], 'С': [18, 27, 10], 'Т': [19,
27, 10], 'У': [16, 23, 11], 'Ф': [17, 23, 11], 'Х': [18, 23, 11], 'Ц': [19, 23, 11], 'Ч': [16
, 24, 11], 'Ш': [17, 24, 11], 'Щ': [18, 24, 11], 'Ь': [19, 24, 11], 'Ы': [16, 25, 11], 'Ъ': [
17, 25, 11], 'Э': [18, 25, 11], 'Ю': [19, 25, 11], 'Я': [16, 26, 11], ' ': [17, 26, 11], '.':
[18, 26, 11], ',': [19, 26, 11], '!': [16, 27, 11], '?': [17, 27, 11], ':': [18, 27, 11], ';'
: [19, 27, 11], '-': [16, 23, 12], '+': [17, 23, 12], '=': [18, 23, 12], '*': [19, 23, 12], '
(': [16, 24, 12], ')': [17, 24, 12], '\\': [18, 24, 12], '/': [19, 24, 12], '>': [16, 25, 12]
, '<': [17, 25, 12], '0': [18, 25, 12], '1': [19, 25, 12], '2': [16, 26, 12], '3': [17, 26, 1
2], '4': [18, 26, 12], '5': [19, 26, 12], '6': [16, 27, 12], '7': [17, 27, 12], '8': [18, 27,
12], '9': [19, 27, 12]}
```

Рис. 7. Результат выполнения функции *funk2*
Fig. 7. Executing *funk2* function

```
def funk3(BUKVA, alf, x0, y0, z0, xx, yy, zz):
    ABC = funk2(alf, x0, y0, z0, xx, yy, zz)
    coord = ABC[BUKVA]
    MATRIX_BUKVA = np.array([[coord[0], coord[1]], [coord[2], coord[0]]])
    return MATRIX_BUKVA
```

Листинг 4. Функция *funk3*
Listing 4. *Funk3* function

```
>>> alf = 'АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЬЫЪЭЮЯ .,!?:;-+=*()\/><0123456789'
>>> funk3('A',alf,16,23,10,19,27,12)
array([[16, 23],
       [10, 16]])
```

Рис. 8. Результат выполнения функции *funk3*
Fig. 8. Executing *funk3* function

строкового типа, отвечающий за открытый текст. В первую очередь из входной строки *TEXT* формируем список символов *TEXT_LIST*. Пользуясь функцией *funk3*, получаем список соответствующих им матриц, причем в качестве первого параметра функции *funk3* вводим элементы списка *TEXT_LIST* (листинг 5).

В результате выполнения этой функции получаем список матриц, в котором каждая матрица соответствует одному символу открытого текста. Таким образом, происходит матричное кодирование исходного текста (рис. 9).

Для **декодирования** открытого текста также используются четыре функции:

1) первая функция *funk1* задает список символов алфавита;
2) вторая функция *funk2* создает словарь, в котором каждой букве в соответствие ставится точка в пространстве;
3) третья функция *funk31* находит символ, который соответствует введенной матрице;
4) четвертая функция *funk41* формирует список символов открытого текста.

Разберем каждую функцию подробнее. Первые две повторяются, т. е. это те же самые функции, что и в первой части программного кода. Рассмотрим последние две функции.

Функция *funk31(MATRIX, alf, x0, y0, z0, xx, yy, zz)* в качестве первого параметра использует матрицу, соответствующую одному из символов выбранного алфавита. Следующие параметры отвечают за список символов этого алфавита, координаты начальной и конечной точек. В начале кода функция *funk2* формирует словарь из символов и точек пространства. Затем в переменную *coord_BUKVA* записываются элементы матрицы, соответствующие координатам точки, задающей символ. Далее перебором ключей словаря циклом *for* проверяется равенство значения в словаре и координаты точки, задающей искомую букву (листинг 6).

В результате выполнения этой функции получаем символ, который соответствует заданной матрице (рис. 10).

Вторая функция *funk41(MATRIX_TEXT, alf, x0, y0, z0, xx, yy, zz)* получает в качестве параметра *MATRIX_TEXT* список матриц и преобразует их в символы, т. е. осуществляет декодирование текста. Используем функцию *funk31*, в качестве ее параметра передаем поочередно элементы списка *MATRIX_TEXT*, а также алфавит и начальную и конечную точки (листинг 7).

В результате выполнения этой функции получаем список символов открытого текста (рис. 11).

```
def funk4(TEXT, alf, x0, y0, z0, xx, yy, zz):
    TEXT_LIST = list(TEXT)
    MATRIX_TEXT=[funk3(TEXT_LIST[i],alf,x0,y0,z0,xx,yy,zz) for i in range(0, len
    (TEXT))]
    return MATRIX_TEXT
```

Листинг 5. Функция *funk4*
Listing 5. *Funk4* function

```
>>> alf = 'АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЬЫЪЭЮЯ .,!?:;-+=*()\/><0123456789'
>>> funk4('ПРИЕМ',alf,16,23,10,19,27,12)
[array([[16, 27],
       [10, 16]]), array([[17, 27],
       [10, 17]]), array([[17, 25],
       [10, 17]]), array([[17, 24],
       [10, 17]]), array([[17, 26],
       [10, 17]])]
```

Рис. 9. Результат выполнения функции *funk4*
Fig. 9. Executing *funk4* function

 https://doi.org/10.21603/2542-1840-2023-7-4-443-456

```
def funk31(MATRIX,alf,x0,y0,z0,xx,yy,zz):
    ABC = funk2(alf,x0,y0,z0,xx,yy,zz)
    coord_BUKVA = [MATRIX[0][0],MATRIX[0][1],MATRIX[1][0]]
    BUKVA = ' '
    for key in ABC:
        if ABC[key] == coord_BUKVA:
            BUKVA = key
    return BUKVA
```

Листинг 6. Функция *funk31*
Listing 6. *Funk31* function

```
>>> alf = 'АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЬЫЪЭЮЯ .,!?:;-+=*()\/><0123456789'
>>> funk31(np.array([[16,27],[10,16]]),alf,16,23,10,19,27,12)
'П'
```

Рис. 10. Результат выполнения функции *funk31*
Fig. 10. Executing *funk31* function

```
def funk41(MATRIX_TEXT,alf,x0,y0,z0,xx,yy,zz):
    TEXT=[funk31(MATRIX_TEXT[i],alf,x0,y0,z0,xx,yy,zz) for i in range(len
    (MATRIX_TEXT))]
    return TEXT
```

Листинг 7. Функция *funk41*
Listing 7. *Funk41* function

```
>>> alf = 'АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЬЫЪЭЮЯ .,!?:;-+=*()\/><0123456789'
>>> funk41([np.array([[16, 27],[10, 16]]), np.array([[17, 27],[10, 17]]),
    np.array([[17, 25],[10, 17]]), np.array([[17, 24],[10, 17]]), np.array
    ([[17, 26],[10, 17]])],alf,16,23,10,19,27,12)
['П', 'Р', 'И', 'Е', 'М']
```

Рис. 11. Результат выполнения функции *funk41*
Fig. 11. Executing *funk41* function

Программная реализация: матричные *k*-граммы

Ранее была обозначена возможность использования матричных *k*-грамм в шифровании. Далее будет представленна ее программная реализация. Первая функция *funkKGramm* отвечает за создания списка матричных *k*-грамм открытого текста. Вторая функция *funkKGramm2* возвращает строку открытого текста. Разберем каждую функцию подробнее.

Функция *funkKGramm(TEXT, k, p)* принимает три параметра. Параметр *TEXT* отвечает за открытый текст, *k* – за длину *k*-граммы, *p* – за модуль дальнейшего шифрования. Первым действием вызывается функция *funk4*, которая переводит исходный текст в список матриц. Далее запускается цикл *while*, который добавляет символы в конец открытого текста, если их не хватает для формирования последней *k*-граммы. Перед циклом for задаются начальные значения для списка *k*-грамм, счетчика и матрицы, отвечающей за каждую *k*-грамму в отдельности. Сам цикл *for* пробегается по всем элементам списка *MATRIX_TEXT* и, проверяя условие, формирует *k*-гамму. На последнем этапе формирования *k*-граммы идет переход в блок *else*, в нем *k*-грамма досчитывается и записывается в список *K_GRAMM*, после чего происходит обнуление счетчика и матрицы, отвечающей за каждую *k*-грамму отдельно (листинг 8).

В результате выполнения этой функции получаем список матричных k-грамм (рис. 12).

Функция *funkKGramm2(K_GRAMM, k, p)* также принимает три параметра. Параметр *K_GRAMM* отвечает за шифр-текст, *k* – за длину *k*-граммы, *p* – за модуль дальнейшего дешифрования. Сначала объявляем пустой список *MATRIX_TEXT*, который будет отвечать за список шифрующих матриц каждого символа отдельно. Далее цикл *for* перебирает элементы списка *K_GRAMM*, внутри цикла объявляется переменная *k_gramm*, которой присваивается значение текущего *x*. В этом цикле пробегается еще один цикл *for* по переменной *i*, меняя ее значение от 1 до *k* + 1. Внутри второго цикла *for* проверяется условие и определяется значение матрицы (численный эквивалент символа открытого текста) по соответствующей формуле. После этого полученное значение добавляется в список *MATRIX_TEXT*, и значение переменной *k_gramm* пересчитывается. Когда переменная *i* принимает значение *k*, происходит переход в блок *else*, где переменная *k_gramm* добавляется в конец списка *MATRIX_TEXT*. Сформировав полный список *MATRIX_TEXT* (его длина будет равна длине открытого текста), применяется функция *funk4*, приняв полученный список как параметр. В итоге получается список символов открытого текста, который записывается в переменную *TEXT*. Далее метод *' '.join()* превращает список в строку (листинг 9).

В результате выполнения этой функции получаем строку открытого текста (рис. 13).

BULLETIN
KEMEROVO STATE UNIVERSITY
HUMANITIES AND SOCIAL SCIENCES

```
def funkKGramm(TEXT,k,p):
    MATRIX_TEXT=funk4(TEXT)
    i = 0
    while len(MATRIX_TEXT) % k != 0:
        MATRIX_TEXT.append(MATRIX_TEXT[i])
        i += 1
    K_GRAMM=[]
    i = 1
    k_gramm = [[0,0], [0,0]]
    for x in MATRIX_TEXT:
        if i%k != 0:
            k_gramm += (p**(k-i))*x
            i += 1
        else:
            k_gramm +=  x
            K_GRAMM.append(k_gramm)
            k_gramm = [[0,0], [0,0]]
            i = 1
    return K_GRAMM
```

Листинг 8. Функция funkKGramm
Listing 8. Funkkgramm function

```
>>> funkKGramm('НЕ МУДРСТВУЯ ЛУКАВО ',2,47)
[array([[ 863, 1246],
        [ 480,  863]]), array([[ 816, 1248],
        [ 527,  816]]), array([[ 768, 1105],
        [ 527,  768]]), array([[ 817, 1296],
        [ 480,  817]]), array([[ 911, 1292],
        [ 480,  911]]), array([[ 768, 1107],
        [ 528,  768]]), array([[ 815, 1248],
        [ 527,  815]]), array([[ 771, 1106],
        [ 527,  771]]), array([[ 770, 1104],
        [ 480,  770]]), array([[ 910, 1248],
        [ 481,  910]])]
```

Рис. 12. Результат выполнения функции *funkKGramm*
Fig. 12. Executing *funkKGramm* function

```
def funkKGramm2(K_GRAMM,k,p):
    MATRIX_TEXT = []
    i = 1
    for x in K_GRAMM:
        k_gramm = x
        for i in range(1, k + 1):
            if i % k != 0:
                MATRIX_TEXT.append(k_gramm//(p**(k-i)))
                k_gramm = k_gramm - k_gramm//(p**(k-i))*(p**(k-i))
            else:
                MATRIX_TEXT.append(k_gramm)
    TEXT = funk41(MATRIX_TEXT)
    return ' '.join(TEXT)
```

Листинг 9. Функция *funkKGramm2*
Listing 9. *FunkKGramm2* function

```
>>> K_GRAMM=[np.array([[ 863, 1246],[ 480,  863]]),np.array([[ 816, 1248],[ 527,  816]]),np.array([[
    768, 1105],[ 527,  768]]),np.array([[ 817, 1296],[ 480,  817]]),np.array([[ 911, 1292],[ 480,  91
    1]]),np.array([[ 768, 1107],[ 528,  768]]),np.array([[ 815, 1248],[ 527,  815]]),np.array([[ 771,
    1106],[ 527,  771]]),np.array([[ 770, 1104],[ 480,  770]]),np.array([[ 910, 1248],[ 481,  910]])]
>>> funkKGramm2(K_GRAMM,2,47)
'НЕ МУДРСТВУЯ ЛУКАВО '
```

Рис. 13. Результат выполнения функции *funkKGramm2*
Fig. 13. Executing *funkKGramm2* function

Учебно-методические материалы

В качестве примеров учебно-методических материалов для уроков математики при изучении методов кодирования и шифрования информации ниже представлены два практических задания по кодированию и декодированию текста, записанного в некотором алфавите. Учащимся предлагается письменно произвести кодирование и декодирование сообщения, а также выполнить проверку, используя программный код на языке Python.

Задание 1. Кодирование открытого текста

Пусть дан алфавит, состоящий из:

- заглавных букв русского алфавита;
- специальных символов «_», «.», «,», «!», «?», «:», «;», «-», «+», «=», «*», «(», «)», «\», «/», «>», «<»;
- цифр: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Используя значения из таблицы 1, соответствующие вашему варианту, закодируйте сообщение. Для решения необходимо построить таблицы, соответствующие горизонтальным плоскостям параллелепипеда, который задается двумя точками (начальной и конечной), и заполнить их заданным алфавитом.

 https://doi.org/10.21603/2542-1840-2023-7-4-443-456

Табл. 1. Варианты практического задания 1
Tab. 1. Task 1: variants

№	Открытый текст	Начальная точка	Конечная точка
1	РОМАШКА	(17,24,11)	(20,28,13)
2	ВАСИЛЁК	(18,25,12)	(21,29,14)
3	ТЮЛЬПАН	(19,26,13)	(22,30,15)

Задание 2. Декодирование шифртекста

Пусть дан алфавит, состоящий из:

- заглавных букв русского алфавита;
- специальных символов «_», «.», «,», «!», «?», «:», «;», «-», «+», «=», «*», «(», «)», «\», «/», «>», «<»;
- цифр: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Используя значения из таблицы 2, соответствующие вашему варианту, декодируйте сообщение. Для решения необходимо построить таблицы, соответствующие горизонтальным плоскостям параллелепипеда, который задается двумя точками (начальной и конечной), и заполнить их заданным алфавитом.

В ходе решения предложенных задач по кодированию текста у учащихся формируются следующие умственные навыки:

- навыки пространственного представления при работе с координатами точек в пространстве,
- вычислительные навыки при составлении координатных матриц и выполнении алгебраических преобразований с ними,
- навыки алгоритмического мышления, выражающиеся в умении действовать по предложенному алгоритму и создавать новые алгоритмы.

Табл. 2. Варианты практического задания 2
Tab. 2. Task 2: variants

№	Шифр-текст	Начальная точка	Конечная точка
1	$\binom{20}{11}\binom{26}{10},\binom{17}{11}\binom{24}{17},\binom{18}{12}\binom{25}{18},$ $\binom{20}{11}\binom{28}{20},\binom{17}{11}\binom{24}{17},\binom{19}{11}\binom{27}{19}.$	(17,24,11)	(20,28,13)
2	$\binom{21}{12}\binom{29}{21},\binom{21}{12}\binom{28}{21},\binom{18}{12}\binom{29}{18},$ $\binom{21}{12}\binom{28}{21},\binom{18}{12}\binom{28}{18},\binom{21}{13}\binom{26}{21}.$	(18,25,12)	(21,29,14)
3	$\binom{20}{13}\binom{30}{20},\binom{19}{14}\binom{29}{19},\binom{20}{13}\binom{26}{20},$ $\binom{20}{13}\binom{28}{20},\binom{20}{13}\binom{29}{21},\binom{19}{13}\binom{26}{19}$	(19,26,13)	(22,30,15)

Заключение

В настоящее время в условиях ФГОС заметна тенденция обновления содержания школьного курса математики. Делаются попытки интегрировать в школьный курс математики различные практико-ориентированные темы и задачи, направленные на актуализацию математических знаний, развитие математической культуры и понимания роли математики в современном мире [23].

В работе были предложены методы кодирования текста защищаемой информации на базе трехмерного параллелепипеда, адаптированные для изучения в рамках школьной математики. Среди них различаются посимвольные и *k*-граммные способы матричного кодирования, которые даже для простых криптографических алгоритмов защиты позволяют повысить криптостойкость ввиду невозможности применения классического частотного анализа текста [6]. Кроме того, они не требуют специальной математической подготовки, поэтому данные методы кодирования могут быть рекомендованы к изучению и применению широкому кругу обучающихся 9–11 классов общеобразовательной школы. При изучении данных алгоритмов кодирования информации у учащихся развиваются пространственное мышление и математическая логика, необходимые для составления трехмерного параллелепипеда (или куба) по заданному алфавиту [24]. На этих примерах они знакомятся с математическим определением матрицы в алгебре, узнают виды шифрования, базовые понятия криптографии, такие как алфавит, открытый текст, шифртекст, *k*-грамма и др. [25], и приобретают актуальные в современном мире навыки защиты текстовой информации.

Включение учебного материала по матричным алгоритмам кодирования информации в школьный курс математики позволит достичь следующие цели:

1. Образовательная: изучить необходимый теоретический материал по элементам линейной алгебры, в частности по теме «Матрицы», и сформировать навыки выполнения элементарных операций над матрицами и вычисления определителя матрицы.

2. Развивающая: содействовать развитию пространственного представления и вычислительных навыков у учащихся, развивать более высокий уровень математической культуры, творческий и прикладной характер мышления.

3. Воспитательная: воспитывать внимательное и ответственное отношение к текстовой информации, умение применять теоретические математические знания на практике для ее защиты.

4. Познавательная: мотивировать учащихся к дальнейшему изучению и практическому применению методов защиты информации.

5. Методическая: способствовать актуализации курса математики в современной школе.

При изучении методов кодирования и комбинированных методов криптографической защиты в школьном курсе математики учащиеся смогут не только расширить свой математический кругозор, но и повысить уровень своей личной информационной безопасности, погрузиться в прикладную часть математики и, возможно, в дальнейшем связать свою профессию с обеспечением безопасности в области информационных технологий. Приведенный в работе программный код, реализующий рассмотренные способы кодирования текста на популярном языке программирования Python, позволяет быстро применить изученные алгоритмы на практике.

Конфликт интересов: Авторы заявили об отсутствии потенциальных конфликтов интересов в отношении исследования, авторства и / или публикации данной статьи.
Conflict of interests: The authors declared no potential conflict of interests regarding the research, authorship, and / or publication of this article.

Критерии авторства: Авторы в равной степени участвовали в подготовке и написании статьи.
Contribution: All the authors contributed equally to the study and bear equal responsibility for information published in this article.

Литература / References

1. Деза Е. И., Котова Л. В., Лебедева Е. С. Теория и практика обучения школьников и студентов СПО основам защиты информации. *Наука и школа*. 2020. № 1. С. 139–153. [Deza E. I., Kotova L.V., Lebedeva E. S. Theory and practice of teaching school and university students the basics of information security. *Science and School*, 2020, (1): 139–153. (In Russ.)] https://elibrary.ru/jpznhs
2. Дегтярева А. А., Пчелинцева Н. В., Макова Н. Е. Математические основы криптологии. *Наука и Образование*. 2020. Т. 3. № 2. [Degtyareva A. A., Pchelintseva N. V., Makova N. E. Mathematical foundations of cryptology. *Nauka i Obrazovanie*, 2020, 3(2). (In Russ.)] URL: http://opusmgau.ru/index.php/see/article/download/1789/1788/ (accessed 2 Jun 2023). https://elibrary.ru/jmndbw
3. Лепшокова А. Р. Разработка интерактивного приложения для наглядного представления шифра Цезаря. *Актуальные проблемы методики обучения информатике и математике в современной школе*: Междунар. науч.-практ. интернет-конф. (Москва, 24 апреля – 12 мая 2020 г.) М.: МПГУ, 2020. С. 493–498. [Lepshokova A. R. Development of an interactive application for visual representation of Caesar's cipher. *Relevant issues of methods of teaching computer science and mathematics in modern school*: Proc. Intern. Sci.-Prac. Internet-Conf., Moscow, 24 Apr – 12 May 2020. Moscow: MPSU, 2020, 493–498. (In Russ.)] https://elibrary.ru/cidefx
4. Коблиц Н. Курс теории чисел и криптографии. М.: ТВП, 2001. 254 с. [Koblits N. *Course of number theory and cryptography*. Moscow: TVP, 2001, 254. (In Russ.)]
5. Мунерман В. И., Самойлова Т. А. Реализация алгоритма шифрования Хилла на основе алгебры многомерных матриц. *Системы высокой доступности*. 2019. Т. 15. № 1. С. 21–27. [Munerman V. I., Samoilova T. A. Implementation of the Hill cipher algorithm on the algebra multidimensional matrices basis. *Sistemy vysokoi dostupnosti*, 2019, 15(1): 21–27. (In Russ.)] https://elibrary.ru/yhtmfh
6. Свечников С. Н. Частотный анализ при использовании классических криптоалгоритмов. *Инновации в науке и практике*: II Междунар. науч.-практ. конф. (Уфа, 17 апреля 2020 г.) Уфа: Вестник науки, 2020. С. 57–62. [Svechnikov S. N. Frequency analysis when using classical crypto algorithms. *Innovations in science and practice*: Proc. II Intern. Sci.-Prac. Conf., Ufa, 17 Apr 2020. Ufa: Vestnik nauki, 2020, 57–62. (In Russ.)] https://elibrary.ru/mdihus
7. Бабаш А. В., Гузовс Р., Касаткин С. В., Прохоров А. Н., Слимов Н. А. Расширение границ применения методов дешифрования шифра Виженера. *Вопросы кибербезопасности.* 2019. № 5. С. 42–50. [Babash A. V., Guzovs R., Kasatkin S. V., Prokhorov A. N., Slimov N. A. Spreading borders of Vigenere cipher decryption methods. *Voprosy kiberbezopasnosti*, 2019, (5): 42–50. (In Russ.)] https://doi.org/10.21681/2311-3456-2019-5-42-50
8. Златопольский Д. М. Некоторые факты из истории использования двоичной системы счисления в докомпьютерную эпоху. *Математика в школе*. 2021. № 5. С. 61–70. [Zlatopolsky D. M. Some facts from the history of using the binary number system in the pre-computer era. *Matematika v shkole*, 2021, (5): 61–70. (In Russ.)] https://elibrary.ru/aolmuw
9. Стрельцова А. С., Ухваркин С. П., Филимонов В. В. Применение эллиптических кривых в алгоритме Диффи-Хеллмана. *Научный альманах*. 2019. № 1-3. С. 62–64. [Streltsova A. S., Ukhvarkin S. P., Filimonov V. V. Application of elliptic curves in the Diffie-Hellman algorithm. *Science Almanac*, 2019, (1-3): 62–64. (In Russ.)] https://elibrary.ru/vwbmsa
10. Кузнецов А. В., Шишкина Э. Л. Методы алгебраической геометрии в криптографии. Воронеж: ВГУ, 2023. 125 с. [Kuznetsov A. V., Shishkina E. L. *Methods of algebraic geometry in cryptography*. Voronezh: VSU, 2023, 125. (In Russ.)] https://elibrary.ru/tlxomb

11. Гулевич С. А. Общие сведения о современной криптографии и подходах к ее изучению. *Ratio et Natura*. 2022. № 2. [Gulevich S. A. General information about modern cryptography and approaches to its study. *Ratio et Natura*, 2022, (2). (In Russ.)] URL: https://ratio-natura.ru/sites/default/files/2023-03/obschie-svedeniya-o-sovremennoy-kriptografii-i-podkhodakh-k-ee-izucheniyu.pdf (accessed 29 May 2023). https://elibrary.ru/pdvnht
12. Хайхан Т. Ю., Никулин В. В. Криптографическая защита информации. *Вестник образовательного консорциума Среднерусский университет. Информационные технологии*. 2021. № 1. С. 4–7. [Haikhan T. Yu., Nikulin V. V. Cryptographic protection of information. *Vestnik obrazovatelnogo konsortsiuma Srednerusskii universitet. Informatsionnye tekhnologii*, 2021, (1): 4–7. (In Russ.)] https://elibrary.ru/rvsbpl
13. Кутовая А. С. Матричные алгоритмы криптографической защиты информации. *Фундаментальные и прикладные исследования в физике, математике и информатике*: XVII (XLIX) Междунар. науч. конф. студентов, аспирантов и молодых ученых «Образование, наука, инновации: вклад молодых исследователей». (Кемерово, 21 апреля 2022 г.) Кемерово: КемГУ, 2022. С. 171–174. [Kutovaya A. S. Matrix algorithms of cryptographic information protection. *Fundamental and applied research in physics, mathematics and computer science*: Proc. XVII (XLIX) Intern. Sci. Conf. of students, postgraduates and young scientists "Education, science, innovation: contribution of young researchers", Kemerovo, 21 Apr 2022. Kemerovo: KemSU, 2022, 171–174. (In Russ.)] https://elibrary.ru/giygkv
14. Кузнецова В. Ю. Обеспечение компетентности российских школьников в вопросах криптографии: анализ целей, возможных подходов и технологий, средств их программной поддержки. *Прикаспийский журнал: управление и высокие технологии*. 2019. № 2. С. 163–170. [Kuznetsova V. Yu. The ensuring competence of Russian students in cryptography issues: Analysis of purposes, possible approaches, means of their software support. *Caspian Journal: Control and High Technologies*, 2019, (2): 163–170. (In Russ.)] https://elibrary.ru/wximcq
15. Гредасова Н. В., Корешникова М. А., Желонкина Н. И., Корчемкина Л. В., Полищук Е. Г., Иванов В. М., Андреева И. Ю. Линейная алгебра. Екатеринбург: УрФУ, 2019. 88 с. [Gredasova N. V., Koreshnikova M. A., Zhelonkina N. I., Korchemkina L. V., Polishchuk E. G., Ivanov V. M., Andreeva I. Iu. *Linear algebra*. Ekaterinburg: UrFU, 2019, 88. (In Russ.)] https://elibrary.ru/wxmlrk
16. Бодров Е. Н. Методическая разработка урока по дисциплине «элементы высшей математики»: алгебра матриц. *Преподавание математики в школах Тверского региона*. 2020. С. 69–78. [Bodrov E. N. Methodical development of a lesson on the discipline "elements of higher mathematics": matrix algebra. *Prepodavanie matematiki v shkolakh Tverskogo regiona*, 2020, 69–78. (In Russ.)] https://elibrary.ru/rqufvm
17. Бабенко А. С., Матыцина Т. Н. Научные основы школьного курса математики. Алгебра. Кострома: КГУ, 2022. 86 с. [Babenko A. S., Matytsina T. N. *Scientific bases of the school course of mathematics. Algebra*. Kostroma: KSU, 2022, 86. (In Russ.)]
18. Кутовая А. С. Элективный курс «Основы криптографии» для учащихся 8–9 классов с углубленным изучением математики. *Фундаментальные и прикладные исследования в физике, химии, математике и информатике*: XVIII (L) Междунар. науч. конф. студентов, аспирантов и молодых ученых «Образование, наука, инновации: вклад молодых исследователей». (Кемерово, 20 апреля 2023 г.) Кемерово: КемГУ, 2023. С. 113–116. [Kutovaya A. S. Elective course in Basic Cryptography for students of 8–9 grades with advanced study of mathematics. *Fundamental and applied research in physics, chemistry, mathematics and computer science*. Proc. XVIII (L) Intern. Sci. Conf. of students, postgraduates, and young scientists: Education, science, and innovation: contribution of young researchers, Kemerovo, 20 Apr 2023. Kemerovo: KemSU, 2023, 113–116. (In Russ.)] https://elibrary.ru/dbepyw
19. Боровков А. С. Профильное обучение как фактор формирования готовности школьника к выбору профессии. *Педагогика и психология современного образования: теория и практика*: 73-я науч.-практ. конф. «Чтения Ушинского». (Ярославль, 5–6 марта 2019 г.) Ярославль: ЯГПУ им. К. Д. Ушинского, 2019. С. 186–189. [Borovkov A. S. Profile training as a factor in the formation of schoolchildren’s readiness to choose a profession. *Pedagogy and psychology of modern education: theory and practice*: Proc. 73 Sci.-Prac. Conf. "Readings of Ushinsky", Yaroslavl, 5–6 Mar 2019. Yaroslavl: YaSPU named after Ushinsky K. D., 2019, 186–189. (In Russ.)] https://elibrary.ru/unnrwr
20. Гришков Д. Ю., Аусилова Н. М. Язык высокого уровня программирования Python. *Наука и реальность*. 2022. № 1. С. 114–117. [Grishkov D. Yu., Ausilova N. M. High-level programming language Python. *Science & Reality*, 2022, (1): 114–117. (In Russ.)] https://elibrary.ru/okojbn
21. Бухаров Т. А., Нафикова А. Р., Мигранова Е. А. Обзор языка программирования Python и его библиотек. *Colloquium-journal*. 2019. № 3-1. С. 23–25. [Bukharov T. A., Nafikova A. R. Migranova E. A. An overview of the Python programming language and its libraries. *Colloquium-journal*, 2019, (3-1): 23–25. (In Russ.)] https://elibrary.ru/yxowbv

22. Пылов П. А., Протодьяконов А. В. Использование и представление массивов в библиотеке NumPy. *Инновации. Наука. Образование*. 2020. № 23. С. 258–266. [Pylov P. A., Protodiakonov A. V. Using and representing arrays in the NumPy library. *Innovatsii. Nauka. Obrazovanie*, 2020, (23): 258–266. (In Russ.)] https://elibrary.ru/naiifo
23. Зверева Л. Г., Корманенко Н. В., Кузнецова Ю. С. Современные тенденции развития методики обучения математике. *The scientific heritage*. 2019. № 40. С. 16–18. [Zvereva L.G., Kormanenko N. V., Kuznetsova U. S. Current trends in the development of teaching mathematics. *The scientific heritage*, 2019, (40): 16–18. (In Russ.)] https://elibrary.ru/lwekma
24. Борбоева Г. М. Роль и место изображений геометрических фигур в формировании пространственного мышления студентов. *Наука, новые технологии и инновации Кыргызстана*. 2019. № 11. С. 186–191. [Borboeva G. M. Role and place of images of geometric figures in the formation of spatial thinking students. *Science, New technologies and Innovations in Kyrgyzstan*, 2019, (11): 186–191. (In Russ.)] https://elibrary.ru/txkpqi
25. Земор Ж. Курс криптографии. М.-Ижевск: Регулярная и хаотическая динамика, 2019. 256 с. [Zémor G. *Course of cryptography*. Moscow-Izhevsk: R&C Dynamics, 2019, 256. (In Russ.)]