

МИНИСТЕРСТВО НАУКИ И ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Московский государственный технический университет им. Н.Э. Баумана

Российский Университет Транспорта

Министерство образования Республики Беларусь

Полоцкий государственный университет



ВОЛОСОВА Н.К. (Московский государственный технический университет им. Н.Э. Баумана – Национальный исследовательский университет);

ВОЛОСОВ К.А., ВОЛОСОВА А.К. (Российский Университет Транспорта);

ПАСТУХОВ Д.Ф., ПАСТУХОВ Ю.Ф. (Полоцкий университет);

СЕЛИВАНОВ В.А. (Полоцкий университет)

АЛГЕБРАИЧЕСКИЕ МЕТОДЫ ШИФРОВАНИЯ

Учебное пособие к лекционным и практическим занятиям
для студентов специальности

1-98 01 01 Компьютерная безопасность

Москва

2023

УДК 519.66

Рецензент:

Карлов Михаил Иванович,

защитил диссертацию кандидата физико-математических наук на Механико-математическом факультете Московского университета им. М.В. Ломоносова

Волосова Н.К., Волосов К.А., Волосова А.К., Пастухов Д.Ф., Пастухов Ю.Ф., Селиванов В.А.

Алгебраические методы шифрования: Учебное пособие (четвертое издание). – Москва. 2023.-52с.

В учебном пособии рассмотрено два примера преобразований $Z_p \rightarrow Z_p \times Z_p$ при шифровании и $Z_p \leftarrow Z_p \times Z_p$ при дешифровании. А также пример отображения $Z_p \times Z_p \times Z_p \rightarrow Z_p \times Z_p \times Z_p$ при шифровании и обратно $Z_p \times Z_p \times Z_p \leftarrow Z_p \times Z_p \times Z_p$ при дешифровании с использованием аффинного преобразования. Один пример отображения $Z_p \rightarrow Z_p \times Z_p \times Z_p$, $Z_p \times Z_p \times Z_p \rightarrow Z_p$. Предложены методы шифрования ростками аналитических функций. Обобщен метод группового кодирования в поле остатков Z_2 на поле остатков группы Z_p с простым модулем p . Третья часть посвящена матричному методу шифрования в группе элементов матриц из Z_p . В восьмом примере рассмотрено линейное пространство правых квазипрямоугольных матриц, а также правила сложения и умножения квазипрямоугольных матриц и алгоритм шифрования с помощью таких матриц. Программы к примерам, написанные на языке FORTRAN(C++), можно использовать в качестве ядра для других программ.

Для студентов университетов, педагогических вузов, а также для студентов технических вузов, преподавателей, инженеров, программистов использующих в своей практической деятельности математические методы шифрования.

УДК 519.66

Оформление «Российский Университет Транспорта», 2023

Оглавление

Введение	4
1. Пример 1 (Algebra 1).	6
2. Пример 2 (Algebra 2).	9
3. Пример 3 (Algebra 3).	13
4. Пример 4 (Algebra 4).	17
5. Пример 5 (Algebra 5).	23
6. Пример 6 (Algebra 6).	26
7. Пример 7 (Algebra 7).	28
8. Пример 8 (Algebra 8).	35
9. Литература	49

ВВЕДЕНИЕ

Под алгебраическим кодированием будем понимать применение алгебры многочленов (кольца многочленов) нескольких переменных с коэффициентами и значениями переменных из множества целочисленных остатков по модулю простого числа p Z_p . Иногда при шифровании с использованием многочленов важно знать нули многочлена, чтобы не делить на нуль при шифровании или дешифровании. Российский математик Алексей Михайлович Матиясевич решил девятую проблему Гильберта. То есть показал, что не существует конечного алгоритма для решения алгебраического полинома нескольких переменных в целых числах. Поэтому возможность применения или запрета заданного набора ключей приходится проверять перед шифрованием программой. В учебном пособии мы рассмотрим три примера алгебраического шифрования.

При шифровании над полем целых остатков Z_p с фиксированными ключами обычно используют отображение множества в себя $Z_p \rightarrow Z_p$, а при дешифровании обратное отображение $Z_p \leftarrow Z_p$. Здесь мы также рассмотрим два примера преобразований $Z_p \rightarrow Z_p \times Z_p$ при шифровании и обратно $Z_p \leftarrow Z_p \times Z_p$ при дешифровании. А также пример взаимно-однозначного аффинного отображения $Z_p \times Z_p \times Z_p \rightarrow Z_p \times Z_p \times Z_p$ при шифровании и обратно $Z_p \times Z_p \times Z_p \leftarrow Z_p \times Z_p \times Z_p$ при дешифровании. Идея третьего примера принадлежит Волосовой Наталье Константиновне.

Второй раздел учебного пособия воплощает идею применения ростка аналитической функции (суммы конечного числа разложения ее в ряд Тейлора – степень последнего слагаемого ряда является ключом шифра) для шифрования текстовых данных над полем целочисленных остатков по простому модулю. Несмотря на то, что применяются аналитические функции, используя целочисленные аргументы ростка функции и простое число p , можно все алгебраические операции корректно проводить на множестве целых остатков группы Z_p . Во втором разделе рассмотрены примеры отображения $Z_p \rightarrow Z_p \times Z_p \times Z_p$ при шифровании и обратного преобразования при дешифровании $Z_p \times Z_p \times Z_p \rightarrow Z_p$ ростками элементарных функций.

Третий раздел обобщает идею профессора В.В. Лидовского из известной монографии "Теория информации"[2] для группового кодирования матрицами с элементами из поля Z_2 до идеи кодирования квадратными матрицами с элементами из поля $Z_p = \{0,1,2,\dots, p-1\}$ (где p произвольное простое число) прямоугольных матриц символьных сообщений с элементами из группы $Z_p = \{0,1,2,\dots, p-1\}$. Квадратная матрица-ключ должна иметь ненулевую главную диагональ для упрощения алгоритма поиска обратной матрицы.

Известно, что квадратные матрицы-ключи с элементами из поля $Z_p = \{0,1,2,\dots, p-1\}$ образуют кольцо квадратных матриц[1], то есть группу по сложению с нейтральным элементом (нулевой матрицей) на множестве элементов $Z_p = \{0,1,2,\dots, p-1\}$ и полугруппу по умножению с нейтральным элементом – единичной матрицей.

В восьмом примере мы применили для шифрования текста линейное пространство квазипрямоугольных матриц. Корректно определена сумма и произведение квазипрямоугольных матриц. Также рассмотрена алгебраическая структура кольца на множестве квазиквадратных матриц. Все программы к примерам в пособии написаны на языке FORTRAN[26]либо на C++, который является одним из распространенных языков в последнее время, поэтому программы на языке C++ могут принести студентам наибольшую пользу.

Раздел 1

Методы шифрования с помощью непрерывных функций и уравнений в частных производных[3],[4],[28] существенно отличаются от методов, основанных на алгебраической теории чисел. Различие заключается не только в непрерывности или дискретности применяемых функций и их аргументов, но и в алгебраических конструкциях, которыми наделены множества в алгебраических методах шифрования, например, кольцо квадратных матриц с элементами из группы Z_p . Алгебраическая теория чисел питает раздел прикладной математики, называемый компьютерной безопасностью. Можно привести несколько последних работ алгебраистов в области прикладной криптографии, например, в работах [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [29]. В разделе 1 учебного пособия используются известные формулы разложения алгебраических полиномов на множители для шифрования текстовых данных над полем целочисленных остатков в группе Z_p .

Пример 1(Algebra 1).

Запишем известную алгебраическую формулу

$$y^n - x^n = (y-x)(y^{n-1} + y^{n-2}x + \dots + yx^{n-2} + x^{n-1}) = (y-x) \sum_{i=0}^{n-1} x^i y^{n-1-i} \quad (1)$$

Пусть целые числа x, n ключи шифрования, y – номер текущего символа в сообщении по таблице ASCII. Введем обозначения

$$R(x, y, n) = y^n - x^n, Q(x, y, n) = \sum_{i=0}^{n-1} x^i y^{n-1-i} \Leftrightarrow (y-x) \stackrel{(1)}{=} \frac{R(x, y, n)}{Q(x, y, n)} \Leftrightarrow y = x + \frac{R(x, y, n)}{Q(x, y, n)}, Q(x, y, n) \neq 0 \quad (2)$$

В формуле(2) исключаются все нули - корни уравнения $y: Q(x, y, n) = 0$ при заданных целых ключах x, n . Рассмотрим алгебраические преобразования(2) на множестве целых положительных остатков Z_p :

Шифром формулой(1) с ключами x, n символа y назовем пару целых чисел $(R(x, y, n) \pmod p; Q(x, y, n) \pmod p)$

$$(R(x, y, n) \pmod p; Q(x, y, n) \pmod p) = \left((y^n - x^n) \pmod p, \sum_{i=0}^{n-1} x^i y^{n-1-i} \pmod p \right), y \notin Y_0 : Q(x, y, n) = 0 \quad (3)$$

$$y \equiv x \pmod p + R(x, y, n) \pmod p \cdot Q^{-1}(x, y, n) \pmod p \mid Q^{-1}(x, y, n) \pmod p \cdot Q(x, y, n) \equiv 1 \pmod p \quad (4)$$

Итак, алгоритм шифрования-дешифрования формулой (1) можно разбить на два случая:

1) $y \notin Y_0 : Q(x, y, n) \neq 0$. Шифрование по формуле(3), дешифрование по формуле(4).

2) $y \in Y_0 : Q(x, y, n) = 0$. Шифром назовем пару чисел $(R(x, y) \equiv x + y \pmod p; Q(x, y) \equiv y - x \pmod p)$ (5)

Дешифрование проводим по формуле(6), отметим, что число $2^{-1} \pmod p$ существует, если p простое.

$$y \equiv R(x, y) + Q(x, y) \equiv x + y + y - x \pmod p = 2y \pmod p \Leftrightarrow y = (R(x, y) + Q(x, y)) \pmod p \cdot 2^{-1} \pmod p \quad (6)$$

Усложним формулы (3)-(6), введем дополнительные целые ключи a, b и множитель $a^b \in N$

1) $y \notin Y_0 : Q(x, y, n) \neq 0$. Шифрование проводим по формуле(7), дешифрование по формуле(8).

$$(R(x, y, n) \pmod p; Q(x, y, n) \pmod p) = \left(a^b \cdot (y^n - x^n) \pmod p, a^b \cdot \sum_{i=0}^{n-1} x^i y^{n-1-i} \pmod p \right), y \notin Y_0 : Q(x, y, n) = 0 \quad (7)$$

$$y \equiv x \pmod p + R(x, y, n) \pmod p \cdot Q^{-1}(x, y, n) \pmod p \mid Q^{-1}(x, y, n) \pmod p \cdot Q(x, y, n) \equiv 1 \pmod p \quad (8)$$

2) $y \in Y_0 : Q(x, y, n) = 0$. Шифром назовем пару чисел ($R(x, y) \equiv x + y \pmod{p}$; $Q(x, y) \equiv y - x \pmod{p}$) (9)

Дешифрование проводим по формуле(10), p простое.

$$y = (R(x, y) + Q(x, y)) \cdot 2^{-1} \pmod{p} \quad (10)$$

Ниже написана программа с использованием алгоритма(7)-(10) на языке FORTRAN, в которой шифруется символьная фраза $s = \text{"Polotsk State University 1234567890"}$ с ключами $a=1119; b=131; n=10000; x=103$. Простое число $p=257$ выбрано ближайшим простым к мощности клавиатуры ASCII равной 256. Допустимы и большие простые числа, чем 257, однако клавиатура ASCII f не сможет в консоли отобразить все символы шифра для произвольного сообщения, однако декодирование все равно производится верно.

```

program algebra1
integer(8),parameter::n=10000,p=257,len=40
integer(8)::x,y,z1,z2,z3,z4,z00,mas1(len+1),mas2(len+1),mas3(len+1)
integer(8)::a,b
character(len+2)::s
integer(8)::mas(len+2)
s="Polotsk State University 1234567890"
a=1119;b=131
x=103;
do i=1,len
mas(i)=ichar(s(i:i))
print*,s(i:i),mas(i)
enddo
do i=1,len
!print*,"i=",i
y=mas(i)
call f1(x,y,n,p,a,b,z2)
if(z2==0)then
z1=mod(x+y,p)
print*,"***** zero *****"
z2=mod(y-x,p)
if(z1<0)then
z1=z1+p
elseif(z2<0)then
z2=z2+p
endif
!print*,z1,z2
call f3(2,p,z3)
z4=mod((z1+z2)*z3,p)
mas1(i)=z4
mas2(i)=0
elseif(.not.z2==0)then
call f3(z2,p,z3)
call f2(x,y,n,p,a,b,z00)

z4=mod(z3*z00+x,p)
if(z4<0)then
z4=z4+p
end if
mas1(i)=0
mas2(i)=z4

```

```

!print*,z2,z00
endif
mas3(i)=mas1(i)+mas2(i)
print*,"i=",i,mas3(i),char(mas3(i))
enddo
pause

```

```

end program algebra1
subroutine f1(x,y,n,p,a,b,z2)
integer(8)::a,b,d
integer(8)::i,j
integer(8)::z1,z,z2,z0,n,p,x,y
z0=0;d=1
do i=1,b
d=mod(d*a,p)
enddo
do i=0,n-1
z1=1;z=1
do j=1,i
z=mod(z*x,p)
enddo
do j=1,n-i-1
z1=mod(z1*y,p)
enddo
z0=mod((z0+z*z1),p)
enddo
z2=mod(z0*d,p)
if(z2<0)then
z2=z2+p
endif
end subroutine
subroutine f2(x,y,n,p,a,b,z00)
integer(8)::a,b,c,d
integer(8)::i
integer(8)::z1,z2,z00,x,y,n,p
d=1
do i=1,b
d=mod(d*a,p)
enddo
z1=1;z2=1;
do i=1,n
z1=mod(z1*x,p)
z2=mod(z2*y,p)
enddo
z00=mod(z2-z1,p)
if(z00<0)then
z00=z00+p
endif
z00=mod(z00*d,p)
end subroutine
subroutine f3(t,p,i,z0)
integer(8)::t,p,i,z0

```

```

do i=1,p-1
if(mod(i*t,p)==1)then
z0=i
endif
enddo
end subroutine

```

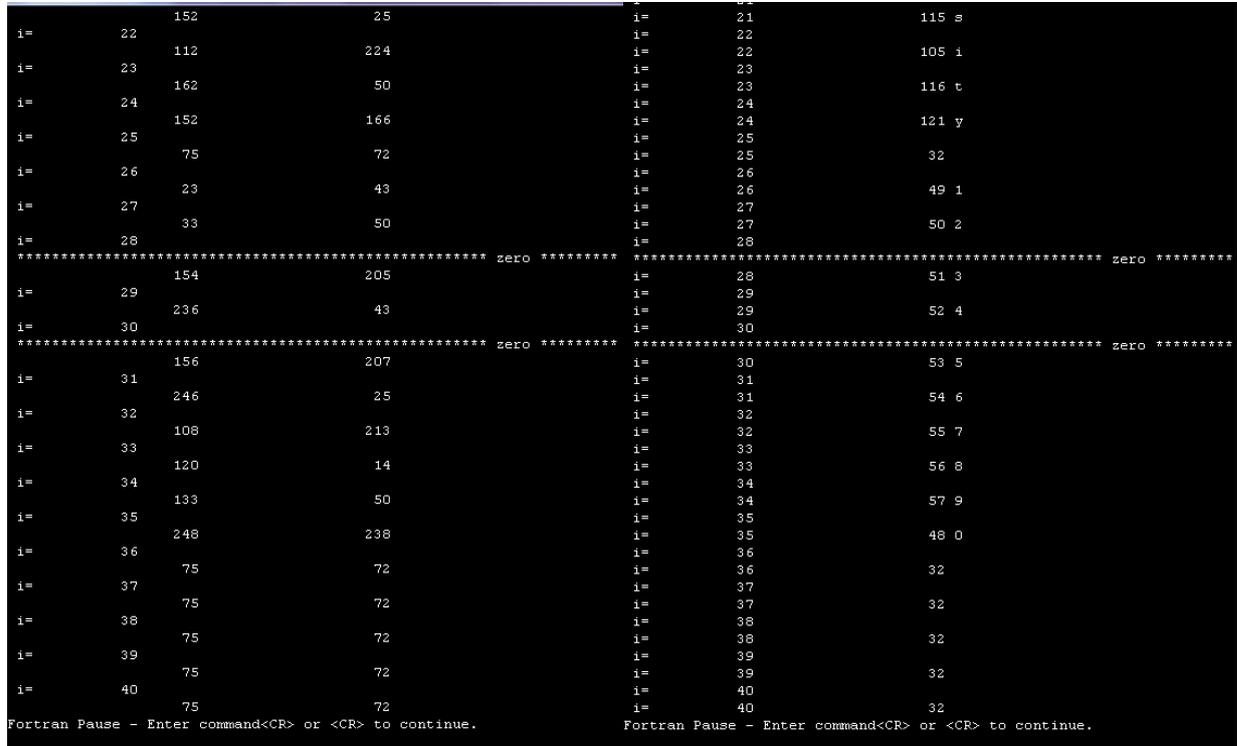


Рис.1

Рис.2

На рисунках 1,2 показан пример шифрования фразы $s="Polotsk State University 1234567890"$ с ключами $a=1119;b=131; n=10000;x=103$. Рисунок 1 представляет шифр конца фразы в консоли, а рисунок 2 конец дешифрованной фразы. Данные рисунки подтверждают, что оба алгоритма программы работают одновременно. Например, символы “3”,”5” с порядковыми номерами $y = \{51;53\} \in Y_0 : \sum_{i=0}^{9999} 103^i y^{9999-i} \pmod{257} \equiv 0$. Поэтому символы “3”,”5” шифруются и дешифруются вторым алгоритмом по формулам(9),(10). Все остальные символы фразы шифруются и дешифруются первым алгоритмом по формулам(7),(8). Отметим, что при больших значениях степени $n=10000$ программа затрачивает на шифрование время 85 с. Дешифрование происходит быстро за микросекунды. Но чем дольше и сложнее алгоритм шифрования, тем сложнее работа крипто-аналитика!

Сделаем оценку пространства ключей и время его прямого перебора. Ключам a,b,x , сопоставим мощность 256, степени n мощность 10000. Тогда имеем мощность пространства ключей $N \approx 256^3 \cdot 10000 = 1.677 \cdot 10^{11}$. Считаем, что в среднем крипто-аналитик затрачивает на подбор одной комбинации ключей время 10^{-3} . Тогда полное время $T \approx 1.677 \cdot 10^{11-3} c = 1.677 \cdot 10^8 c = 1941 \text{ суток} \approx 5,3 \text{ лет}$. При необходимости в алгоритм(7)-(10) можно добавить еще два ключа.

Отметим также, что отдельные массивы – шифры двух алгоритмов на конечном этапе после дешифрования объединяются в один массив (складываются после дешифрования в один массив по правилам арифметики).

Размещая в консоли все записи компактно, из рисунка 3 видно, что все символы фразы $s="Polotsk State University 1234567890"$ с числом знаков 40(лишние символы не

входящие во фразу дешифруются пробелами – с порядковым номером символа 32) декодируются взаимно-однозначно.

```

00001 - [Graphic1]
                                     32
i=      1          80 P
i=      2          111 o
i=      3          108 l
i=      4          111 o
i=      5          116 t
i=      6          115 s
i=      7          107 k
i=      8           32
i=      9          83 S
i=     10          116 t
i=     11          97 a
i=     12          116 t
i=     13          101 e
i=     14           32
i=     15          85 U
i=     16          110 n
i=     17          105 i
i=     18          118 v
i=     19          101 e
i=     20          114 r
i=     21          115 s
i=     22          105 i
i=     23          116 t
i=     24          121 y
i=     25           32
i=     26          49 1
i=     27          50 2
i=     28          51 3
i=     29          52 4
i=     30          53 5
i=     31          54 6
i=     32          55 7
i=     33          56 8
i=     34          57 9
i=     35          48 0
i=     36           32
i=     37           32
i=     38           32
i=     39           32
i=     40           32
Fortran Pause - Enter command<CR> or <CR> to continue.

```

Рис.3. Пример дешифрования программой Algebra 1.

Пример 2(Algebra 2).

Запишем вторую алгебраическую формулу

$$y^{2k+1} + x^{2k+1} = (y+x)(y^{2k} - y^{2k-1}x + \dots + (-1)^{2k-1}yx^{n-2} + (-1)^{2k}x^{2k}) = (y+x)\sum_{i=0}^{2k} x^i y^{2k-i} \quad (11)$$

Пусть целые числа x, n ключи шифрования, y – номер текущего символа в сообщении по таблице ASCII. Введем обозначения

$$R(x, y, n) = y^{2k+1} + x^{2k+1}, Q(x, y, n) = \sum_{i=0}^{2k} x^i y^{2k-i} \Leftrightarrow (y+x)^{(11)} = \frac{R(x, y, n)}{Q(x, y, n)} \Leftrightarrow y = x + \frac{R(x, y, n)}{Q(x, y, n)}, Q(x, y, n) \neq 0 \quad (12)$$

В формуле(12) исключаются все нули - корни уравнения $y: Q(x, y, n) = 0$ при заданных целых ключах x, n . Рассмотрим алгебраические преобразования(12) на множестве целых положительных остатков Z_p :

Шифром формулой(10) с ключами x, n символа y назовем пару целых чисел $(R(x, y, n)(\text{mod } p); Q(x, y, n)(\text{mod } p))$

$$(R(x, y, n)(\text{mod } p); Q(x, y, n)(\text{mod } p)) = \left((y^{2k+1} + x^{2k+1})(\text{mod } p), \sum_{i=0}^{2k} x^i y^{2k-i}(\text{mod } p) \right), y \notin Y_0 : Q(x, y, n) = 0 \quad (13)$$

$$y \equiv x(\text{mod } p) + R(x, y, n)(\text{mod } p) \cdot Q^{-1}(x, y, n)(\text{mod } p) \mid Q^{-1}(x, y, n)(\text{mod } p) \cdot Q(x, y, n) \equiv 1(\text{mod } p) \quad (14)$$

Итак, алгоритм шифрования-дешифрования формулой (1) можно разбить на два случая:

1) $y \notin Y_0 : Q(x, y, n) \neq 0$. Шифрование по формуле(13), дешифрование по формуле(14).

2) $y \in Y_0 : Q(x, y, n) = 0$. Шифром назовем пару чисел $(R(x, y) \equiv x + y(\text{mod } p); Q(x, y) \equiv y - x(\text{mod } p))$ (15)

Дешифрование проводим по формуле(16),отметим, что число $2^{-1}(\text{mod } p)$ существует, если p простое.

$$y \equiv R(x, y) + Q(x, y) \equiv x + y + y - x(\text{mod } p) = 2y(\text{mod } p) \Leftrightarrow y = (R(x, y) + Q(x, y))(\text{mod } p) \cdot 2^{-1}(\text{mod } p) \quad (16)$$

Усложним формулы (13)-(16), введем дополнительные целые ключи a, b и множитель $a^b \in N$

1) $y \notin Y_0 : Q(x, y, n) = 0$. Шифрование проводим по формуле(17), дешифрование по формуле(18).

$$(R(x, y, n)(\text{mod } p); Q(x, y, n)(\text{mod } p)) = \left(a^b \cdot (y^{2k+1} + x^{2k+1})(\text{mod } p), a^b \cdot \sum_{i=0}^{n-1} x^i y^{n-1-i}(\text{mod } p) \right), y \notin Y_0 : Q(x, y, n) = 0 \quad (17)$$

$$y \equiv x(\text{mod } p) + R(x, y, n)(\text{mod } p) \cdot Q^{-1}(x, y, n)(\text{mod } p) \mid Q^{-1}(x, y, n)(\text{mod } p) \cdot Q(x, y, n) \equiv 1(\text{mod } p) \quad (18)$$

2) $y \in Y_0 : Q(x, y, n) = 0$. Шифром назовем пару чисел $(R(x, y) \equiv x + y(\text{mod } p); Q(x, y) \equiv y - x(\text{mod } p))$ (19)

Дешифрование проводим по формуле(20), p простое.

$$y = (R(x, y) + Q(x, y))(\text{mod } p) \cdot 2^{-1}(\text{mod } p) \quad (20)$$

Ниже написана программа с использованием алгоритма(17)-(20) на языке FORTRAN, в которой шифруется символьная фраза $s = \text{"Polotsk State University 1234567890"}$ с ключами $a=1119; b=131; n=10000; x=103$. Простое число $p=257$ выбрано ближайшим простым к мощности клавиатуры ASCII равной 256. Допустимы и большие простые числа, чем 257, однако клавиатура ASCII f не сможет в консоли отобразить все символы шифра для произвольного сообщения, однако декодирование все равно производится верно.

```

program algebra
integer(8),parameter::n=599,p=257,len=40
integer(8)::x,y,z1,z2,z3,z4,z00,mass1(len+1),mass2(len+1),mass3(len+1)
integer(8)::a,b
character(len+2)::s
integer(8)::mass(len+2)
s="Polotsk State University 1234567890"
a=1119;b=131
x=1;
do i=1,len
mass(i)=ichar(s(i:i))
print*,s(i:i),mass(i)
enddo
do i=1,len
y=mass(i)
call f1(x,y,n,p,a,b,z2)
if(z2==0)then
print*,"error"
z1=mod(x+y,p)
z2=mod(y-x,p)
call f3(2,p,z3)
z4=mod((z1+z2)*z3,p)
mass1(i)=z4
mass2(i)=0
elseif(.not.z2==0)then
call f3(z2,p,z3)
call f2(x,y,n,p,a,b,z00)
z4=mod(z3*z00-x,p)
if(z4<0)then

```

```

z4=z4+p
end if
mas1(i)=0
mas2(i)=z4
endif
mas3(i)=mas1(i)+mas2(i)
print*,"i=",i,mas3(i),char(mas3(i))
enddo
end program algebra
subroutine f1(x,y,n,p,a,b,z2)
integer(8)::a,b,d
integer(8)::i,j
integer(8)::z1,z,z2,z0,n,p,x,y,zz
z0=0;d=1;zz=-1
do i=1,b
d=mod(d*a,p)
enddo
do i=0,n-1
z1=1;z=1
do j=1,i
z=mod(z*x,p)
enddo
do j=1,n-i-1
z1=mod(z1*y,p)
enddo
zz=-zz
z0=mod((z0+z*z1*zz),p)
enddo
z2=mod(z0*d,p)
if(z2<0)then
z2=z2+p
endif
end subroutine
subroutine f2(x,y,n,p,a,b,z00)
integer(8)::a,b,c,d
integer(8)::i
integer(8)::z1,z2,z00,x,y,n,p
d=1
do i=1,b
d=mod(d*a,p)
enddo
z1=1;z2=1;
do i=1,n
z1=mod(z1*x,p)
z2=mod(z2*y,p)
enddo
z00=mod(z2+z1,p)
if(z00<0)then
z00=z00+p
endif
z00=mod(z00*d,p)
end subroutine

```

```

subroutine f3(t,p,z0)
integer(8)::t,p,i,z0
do i=1,p-1
if(mod(i*t,p)==1)then
z0=i
endif
enddo
end subroutine

```

Рис.4. (n=10001) a=1119;b=131; x=103

Рис.5.(n=10000) a=1119;b=131; x=103

На рисунках 3,4 показан пример шифрования фразы s="Polotsk State University 1234567890" с ключами a=1119;b=131; x=103; n=10001(Рис.4),n=10000(Рис.5). Рисунок 4 показывает, что ни при каких символах фразы при заданных ключах знаменатель дроби не содержит нулей $\forall y \in Y_0 : Q(x, y, n) \neq 0$. Рисунки 4, 5 подтверждают, что шифрование формулой (11) возможно только при нечетных значениях степени, программа затрачивает на шифрование время 88 с. Дешифрование происходит значительно быстрее. Но чем дольше и сложнее алгоритм шифрования, тем сложнее работа крипто-аналитика!

Оценку пространства ключей и время его прямого перебора возьмем из примера 1. Мощность пространства $N \approx 256^3 \cdot 10000 = 1.677 \cdot 10^{11}$, за время $T \approx 1.677 \cdot 10^{11} \cdot c = 1.677 \cdot 10^8 c = 194 \text{ суток} \approx 5,3 \text{ лет}$. При необходимости в алгоритм(7)-(10) можно добавить еще два ключа. Как и в примере 1 отдельные массивы – шифры двух алгоритмов на конечном этапе после дешифрования объединяются в один массив (складываются после дешифрования в один массив по правилам арифметики). Из рисунка 4 видно, что символы фразы s="Polotsk State University 1234567890" с числом знаков 40(лишние символы не входящие во фразу дешифруются пробелами – с порядковым номером символа 32) декодируются взаимно-однозначно.

Пример 3(Algebra 3).

$$\text{Обозначим 12 ключей шифрования символами } A = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix}, b = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \quad (21)$$

Для каждой тройки последовательных символов фразы (x, y, z) найдем тройку символов шифрованной фразы (x_1, y_1, z_1) используя линейное (аффинное) отображение $Z_p \times Z_p \times Z_p \rightarrow Z_p \times Z_p \times Z_p$ по формуле (шифрования)

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = A \begin{bmatrix} x \\ y \\ z \end{bmatrix} + b \Leftrightarrow \begin{cases} x_1 = a_{1,1}x + a_{1,2}y + a_{1,3}z + b_1 \\ y_1 = a_{2,1}x + a_{2,2}y + a_{2,3}z + b_2 \\ z_1 = a_{3,1}x + a_{3,2}y + a_{3,3}z + b_3 \end{cases} \quad (22)$$

Тогда, из формулы(22) получим исходную тройку символов

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = A^{-1} \begin{bmatrix} x_1 - b_1 \\ y_1 - b_2 \\ z_1 - b_3 \end{bmatrix} \Leftrightarrow \frac{1}{\Delta} \begin{cases} A_{1,1}(x_1 - b_1) + A_{1,2}(y_1 - b_2) + A_{1,3}(z_1 - b_3) \\ A_{2,1}(x_1 - b_1) + A_{2,2}(y_1 - b_2) + A_{2,3}(z_1 - b_3), \Delta \neq 0 \\ A_{3,1}(x_1 - b_1) + A_{3,2}(y_1 - b_2) + A_{3,3}(z_1 - b_3) \end{cases} \quad (23)$$

Где:

$$\Delta = \begin{vmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{vmatrix} = a_{1,1} \begin{vmatrix} a_{2,2} & a_{2,3} \\ a_{3,2} & a_{3,3} \end{vmatrix} - a_{1,2} \begin{vmatrix} a_{2,3} & a_{2,3} \\ a_{3,1} & a_{3,3} \end{vmatrix} + a_{1,3} \begin{vmatrix} a_{2,1} & a_{2,2} \\ a_{3,2} & a_{3,3} \end{vmatrix} = a_{1,1}(a_{2,2}a_{3,3} - a_{3,2}a_{2,3}) + a_{1,2}(a_{3,1}a_{2,3} - a_{2,3}a_{3,3}) + a_{1,3}(a_{2,1}a_{3,3} - a_{3,2}a_{2,2})$$

$$\begin{cases} A_{1,1} = a_{2,2}a_{3,3} - a_{3,2}a_{2,3}; A_{2,1} = a_{2,3}a_{3,1} - a_{2,1}a_{3,3}; A_{3,1} = a_{2,1}a_{3,2} - a_{3,1}a_{2,2} \\ A_{1,2} = a_{3,2}a_{1,3} - a_{1,2}a_{3,3}; A_{2,2} = a_{1,1}a_{3,3} - a_{1,3}a_{3,1}; A_{3,2} = a_{1,2}a_{3,1} - a_{1,1}a_{3,2} \\ A_{1,3} = a_{1,2}a_{2,3} - a_{1,3}a_{2,2}; A_{2,3} = a_{2,1}a_{1,3} - a_{1,1}a_{2,3}; A_{3,3} = a_{2,2}a_{1,1} - a_{1,2}a_{2,1} \end{cases} \quad (24)$$

Тогда формула шифрования над полем целых остатков Z_p с учетом формулы(22) имеет вид

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} (\text{mod } p) \equiv A \begin{bmatrix} x \\ y \\ z \end{bmatrix} (\text{mod } p) + b (\text{mod } p) \Leftrightarrow \begin{cases} x_1 \equiv a_{1,1}x + a_{1,2}y + a_{1,3}z + b_1 (\text{mod } p) \\ y_1 \equiv a_{2,1}x + a_{2,2}y + a_{2,3}z + b_2 (\text{mod } p) \\ z_1 \equiv a_{3,1}x + a_{3,2}y + a_{3,3}z + b_3 (\text{mod } p) \end{cases} \quad (25)$$

Далее для дешифрования нужно вычислить таблицу вспомогательных чисел согласно(24)

$$\begin{cases} A_{1,1} \equiv a_{2,2}a_{3,3} - a_{3,2}a_{2,3} (\text{mod } p); A_{2,1} \equiv a_{2,3}a_{3,1} - a_{2,1}a_{3,3} (\text{mod } p); A_{3,1} \equiv a_{2,1}a_{3,2} - a_{3,1}a_{2,2} (\text{mod } p) \\ A_{1,2} \equiv a_{3,2}a_{1,3} - a_{1,2}a_{3,3} (\text{mod } p); A_{2,2} \equiv a_{1,1}a_{3,3} - a_{1,3}a_{3,1} (\text{mod } p); A_{3,2} \equiv a_{1,2}a_{3,1} - a_{1,1}a_{3,2} (\text{mod } p) \\ A_{1,3} \equiv a_{1,2}a_{2,3} - a_{1,3}a_{2,2} (\text{mod } p); A_{2,3} \equiv a_{2,1}a_{1,3} - a_{1,1}a_{2,3} (\text{mod } p); A_{3,3} \equiv a_{2,2}a_{1,1} - a_{1,2}a_{2,1} (\text{mod } p) \end{cases} \quad (26)$$

Найти обратное число к матрице системы(25)

$$\Delta^{-1} (\text{mod } p): \Delta^{-1} \cdot (a_{1,1}(a_{2,2}a_{3,3} - a_{3,2}a_{2,3}) + a_{1,2}(a_{3,1}a_{2,3} - a_{2,3}a_{3,3}) + a_{1,3}(a_{2,1}a_{3,3} - a_{3,2}a_{2,2})) \equiv 1 (\text{mod } p) \quad (27)$$

Если определитель матрицы системы(25) сравним с нулем $\Delta \equiv 0 (\text{mod } p)$, то изменить какие-нибудь из 9 ключей матрицы системы шифрования(21). Наконец, запишем формулы дешифрования аналогично формулам(23)

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \equiv \Delta^{-1} (\text{mod } p) \cdot \begin{cases} A_{1,1}(x_1 - b_1) + A_{1,2}(y_1 - b_2) + A_{1,3}(z_1 - b_3) (\text{mod } p) \\ A_{2,1}(x_1 - b_1) + A_{2,2}(y_1 - b_2) + A_{2,3}(z_1 - b_3) (\text{mod } p) \\ A_{3,1}(x_1 - b_1) + A_{3,2}(y_1 - b_2) + A_{3,3}(z_1 - b_3) (\text{mod } p) \end{cases} \Leftrightarrow$$

$$\Leftrightarrow \begin{cases} x \equiv \Delta^{-1}(\bmod p)(A_{1,1}(x_1 - b_1) + A_{1,2}(y_1 - b_2) + A_{1,3}(z_1 - b_3))(\bmod p) \\ y \equiv \Delta^{-1}(\bmod p)(A_{2,1}(x_1 - b_1) + A_{2,2}(y_1 - b_2) + A_{2,3}(z_1 - b_3))(\bmod p), \Delta \neq 0(\bmod p) \\ z \equiv \Delta^{-1}(\bmod p)(A_{3,1}(x_1 - b_1) + A_{3,2}(y_1 - b_2) + A_{3,3}(z_1 - b_3))(\bmod p) \end{cases} \quad (28)$$

Очевидно, что указанную последовательность алгоритма нужно дополнить способом разбиения исходной фразы на тройки символов. Если число символов сообщения не делится на три, то дополнить массив сообщения до ближайшего целого числа кратного трём. Избыточные символы массива сообщения (один либо два) при дешифровании окажутся пробелами с порядковым номером 32 и полезной информации не несут и не искажают исходное сообщение.

Ниже написана программа с использованием алгоритма(25)-(28) на языке FORTRAN, в которой шифруется символьная фраза s="Moscow State University 265 years" с ключами b1=123;b2=66;b3=38;a11=1;a12=2;a13=5;a21=23;a31=4;a22=5;a23=6;a32=7;a33=8. Простое число p=257 выбрано ближайшим простым к мощности клавиатуры ASCII равной 256. Допустимы и большие простые числа, чем 257, однако клавиатура ASCII f не сможет в консоли отобразить все символы шифра для произвольного сообщения, однако декодирование все равно производится верно.

```

program algebra
integer(8),parameter::p=257,kk=12,len=3*kk
integer(8)::x,y,z,z1,z2,z3,z4,z00,mas1(len+1),mas2(len+1),mas3(len+1)
integer(8)::a11,a12,a13,a21,a22,a23,a31,a32,a33,b1,b2,b3
integer(8)::da11,da12,da13,da21,da22,da23,da31,da32,da33,delta
integer(8)::xx,yy,zz,x1,y1,shifr(4,len+1),i,j,k,mas0(3,len+1)
character(len+2)::s
integer(8)::mas(len+2)
s="Moscow State University 265 years"
b1=123;b2=66;b3=38;
a11=3;a12=0;a13=5;a21=13;a31=4;a22=15;a23=7;a32=6;a33=18
da11=mod((a22*a33-a23*a32),p)
da21=mod((a31*a23-a21*a33),p)
da31=mod((a21*a32-a22*a31),p)
da12=mod((a32*a13-a12*a33),p)
da22=mod((a11*a33-a13*a31),p)
da32=mod((a12*a31-a11*a32),p)
da13=mod((a12*a23-a13*a22),p)
da23=mod((-a11*a23+a21*a13),p)
da33=mod((a22*a11-a12*a21),p)
delta=mod(a11*(a22*a33-a23*a32)-a12*(a21*a33-a31*a23)+a13*(a21*a32-a31*a22),p)
call f(delta,p,z3)
print*,mod(delta*z3,p)
if(.not.delta==0)then
print*,"delta=",delta
else
print*,"viberete drugie kluchi"
print*,"delta=",delta
end if
print*,"delta^-1=",z3
do i=1,len
mas(i)=ichar(s(i:i))
print*,i,s(i:i),mas(i)

```

```

enddo
k=0
print*, "*****shifr*****"
do i=1,len
if(mod(i,3)==1)then
x=mas(i)
k=k+1
elseif(mod(i,3)==2)then
y=mas(i)
k=k+1
elseif(mod(i,3)==0)then
z=mas(i)
k=k+1
endif
if(mod(k,3)==0)then
x1=mod((a11*x+a12*y+a13*z+b1),p)
y1=mod((a21*x+a22*y+a23*z+b2),p)
z1=mod((a31*x+a32*y+a33*z+b3),p)
shifr(1,k/3)=x1
shifr(2,k/3)=y1
shifr(3,k/3)=z1
print*,x1,y1,z1
endif
enddo
print*, "*****deshifr*****"
do i=1,len/3
x=z3*mod((da11*(shifr(1,i)-b1)+da12*(shifr(2,i)-b2)+da13*(shifr(3,i)-b3)),p)
y=z3*mod((da21*(shifr(1,i)-b1)+da22*(shifr(2,i)-b2)+da23*(shifr(3,i)-b3)),p)
z=z3*mod((da31*(shifr(1,i)-b1)+da32*(shifr(2,i)-b2)+da33*(shifr(3,i)-b3)),p)
x=mod(x,p)
y=mod(y,p)
z=mod(z,p)
if(x<0)then
x=x+p
endif
if(y<0)then
y=y+p
endif
if(z<0)then
z=z+p
endif
mas0(1,i)=x
mas0(2,i)=y
mas0(3,i)=z
print*,x,y,z
enddo
print*, "*****pause*****"
!pause
do i=1,len/3
print*,char(mas0(1,i))
print*,char(mas0(2,i))
print*,char(mas0(3,i))

```

```

enddo
end program algebra
subroutine f(t,p,z0)
integer(8)::t,p,i,z0
do i=1,p-1
if(mod(i*t,p)==1)then
z0=i
endif
enddo
end subroutine

```

Оценим мощность пространства ключей. Учитывая, что каждый из 12 ключей можно выбирать независимо из 256 вариантов, то мощность искомого множества равна $N = 256^{12} \approx 7.92 \cdot 10^{28}$. Так как процессор крипто-аналитика не сможет перебирать ключи быстрее чем один набор за 1 такт процессора (одна миллиардная одной секунды), то получим полное время перебора $T \approx 7.92 \cdot 10^{28} \cdot 10^{-9} = 7.92 \cdot 10^{19} s = 2.51 \cdot 10^{12} years$.

На рисунках 6,7 показан пример шифрования фразы “Moscow State University 265 years”. На рисунке 6 видно, что в массиве программы шифр, а также дешифрованный текст хранится тройками в каждой строке, затем указанные векторы переписываются последовательно в один столбец и переводятся в символы таблицей ASCII. На рисунке 7 видно, что исходная фраза полностью дешифруется верно.

```

18 e          101
19 r          114
20 s          115
21 i          105
22 t          116
23 y          121
24           32
25 2          50
26 6          54
27 5          53
28           32
29 y          121
30 e          101
31 a          97
32 r          114
33 s          115
34           32
35           32
36           32
*****shifr*****
158          196          255
244          253          158
28           226          182
148          176          113
255          214          86
172          53           157
219          153          247
117          15           5
24           98           231
210          177          140
218          244          96
122          158          163
*****deshifr*****
77           111          115
99           111          119
32           83           116
97           116          101
32           85           110
105          118          101
114          115          105
116          121          32
50           54           53
32           121          101
97           114          115
32           32           32
*****pause*****

```

Рис.6

```

97           116          101
32           85           110
105          118          101
114          115          105
116          121          32
50           54           53
32           121          101
97           114          115
32           32           32
*****pause*****
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
M
o
s
c
o
w

S
t
a
t
e

U
n
i
v
e
r
s
i
t
y

2
6
5

y
e
a
r
s

```

Рис.7

Число символов равно 35, однако исходный массив расширяется до ближайшего целого кратного 3 -36, то есть 36 символ по счету (не несущий полезной информации) идентифицируется как пробел с порядковым номером 32, что и видно из рисунка 7.

Отметим, что перемножение многочленов в примерах Algebra 1, Algebra 2 можно свести матричному кодированию или групповому кодированию (используется в примере Algebra 3). Матричное кодирование обладает рядом замечательных свойств[1]. Если пространство исходных слов обладает групповыми свойствами, то пространство шифрованных слов также является группой. Множество целых остатков по модулю p Z_p является полем, а операция сложения двух чисел по модулю всегда обратима. Следовательно, пространство слов шифра одинаковой длины образуют группу с операцией сложения по модулю p (в примере Algebra 3). Кроме того, если коэффициенты и аргументы многочленов из группы Z_2 , то минимальное расстояние между шифрованными словами равно минимальному весу слова, отличного от нуля. Весом слова двоичного кода равен числу единиц в его записи или сумме всех его цифр[1].

Раздел 2

В данном разделе мы опишем совершенно новую идею шифрования аналитическими ростками дифференцируемых функций одного аргумента.

Пример 4(Algebra 4).

Обозначим росток элементарной функции $y(x) = \exp(x)$ с n слагаемыми как $R_{\exp(x)}^n(x)$

$$R_{\exp(x)}^n(x) = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \dots + \frac{x^n}{n!}, \exp(x) = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \dots + \frac{x^n}{n!} + O(x^{n+1}) \quad (29)$$

Пусть целое число n и вид функции $y(x)$ ключи шифрования, x – номер текущего символа в сообщении по таблице ASCII(натуральное число $x \in N, 0 \leq x \leq p-1$). Введем обозначения

$$R(x, n) = R_{\exp(x)}^{n-1}(x) = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \dots + \frac{x^{n-1}}{(n-1)!}, Q(x, y, n) = R_{\exp(x)}^n(x) = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \dots + \frac{x^n}{n!}, Z(x, n) = \frac{x^{n-1}}{n!} \quad (30)$$

В формуле(30) $R(x, n), Q(x, y, n), Z(x, y, n)$ определены при всех целых числах. Рассмотрим алгебраические преобразования(30) на множестве целых положительных остатков Z_p

Шифром (формула(31)) с ключами p, n символа y назовем тройку целых чисел $(R(x, n), Q(x, y, n), Z(x, y, n)(\text{mod } p))$

$$(R(x, n), Q(x, y, n), Z(x, y, n)(\text{mod } p)) = \left(R_{y(x)}^{n-1}(x), R_{y(x)}^n(x), \frac{x^{n-1}}{n!} \right) (\text{mod } p) \quad (31)$$

То есть

$$R(x, n) = R_{\exp(x)}^{n-1}(x) = \left(1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \dots + \frac{x^{n-1}}{(n-1)!} \right) (\text{mod } p) = \sum_{i=0}^{n-1} x^i \cdot (i!)^{-1} (\text{mod } p), (i!)^{-1} : (i!)^{-1} (i!) \equiv 1 (\text{mod } p) \quad (32)$$

$$Q(x, n) = R_{\exp(x)}^n(x) = \left(1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \dots + \frac{x^n}{n!} \right) (\text{mod } p) = \sum_{i=0}^n x^i \cdot (i!)^{-1} (\text{mod } p), (i!)^{-1} : (i!)^{-1} (i!) \equiv 1 (\text{mod } p) \quad (33)$$

$$Z(x, n) = \frac{x^{n-1}}{n!} (\text{mod } p) = x^{n-1} \cdot (n!)^{-1} (\text{mod } p), (n!)^{-1} : (n!)^{-1} \cdot (n!) \equiv 1 (\text{mod } p) \quad (34)$$

Из определения шифра формулы(32),(33),(34), видно, что они корректно определены при любых целых значениях $x, n, n < p$

Дешифрование проводим по формуле(35)

$$x \equiv (Q(x, n) - R(x, n)) Z(x, y, n)^{-1} (\text{mod } p), (Z(x, y, n))^{-1} : (Z(x, y, n))^{-1} \cdot Z(x, y, n) \equiv 1 (\text{mod } p) \quad (35)$$

Из формул(34),(35)следует, что для корректности необходимо условие $1 \leq x \leq p-1$.
Далее следует программный код нашего алгоритма (29)-(35)
program algebra

```

integer(8),parameter::n=8,p=257,len=45
integer(8)::x,y,z0,z1,z2,z3,z4,z00,mas1(len+1),mas2(len+1),mas3(len+1)
integer(8)::a,b,t,i,j,shifr(3,len+1)
character(len+2)::s
integer(8)::mas(len+2)
s="Undergraduate of Polotsk State University 2020"
do i=1,len
mas(i)=ichar(s(i:i))
print*,s(i:i),mas(i)
enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
pause
do j=1,len
x=mas(j)
a=1
do i=1,n-1
call factor(x,i,p,z0)
a=mod(a+z0,p)
enddo
shifr(1,j)=a
b=1
do i=1,n
call factor(x,i,p,z0)
b=mod(b+z0,p)
enddo
shifr(2,j)=b
call factorial(x,n,p,z0)
shifr(3,j)=z0
print*,shifr(1,j),shifr(2,j),shifr(3,j)
enddo
pause
do j=1,len
a=shifr(2,j)-shifr(1,j)
call f3(shifr(3,j),p,z0)
b=mod(z0*a,p)
if(b<0)then
b=b+p
endif
print*,b,char(b)
enddo
pause
end program algebra
subroutine factor(x,i,p,z0)
integer(8)::x,i,p,k,z,z1,z0
z=1
do k=1,i
z=mod(z*x,p)
enddo
z1=1
do k=1,i
z1=mod(z1*k,p)
enddo

```

```

call f3(z1,p,z0)
z0=mod(z0*z,p)
end subroutine
subroutine factorial(x,i,p,z0)
integer(8)::x,i,p,k,z,z1,z0
z=1
do k=1,i-1
z=mod(z*k,p)
enddo
z1=1
do k=1,i
z1=mod(z1*k,p)
enddo
call f3(z1,p,z0)
z0=mod(z0*z,p)
end subroutine
subroutine f3(t,p,z0)
integer(8)::t,p,i,z0
do i=1,p-1
if(mod(i*t,p)==1)then
z0=i
endif
enddo
end subroutine

```

На рисунках 8,9 шифруются символы фразы s="Undergraduate of State University 2020"(магистранты Полоцкого государственного университета 2020 года) и дешифруются взаимно-однозначно.

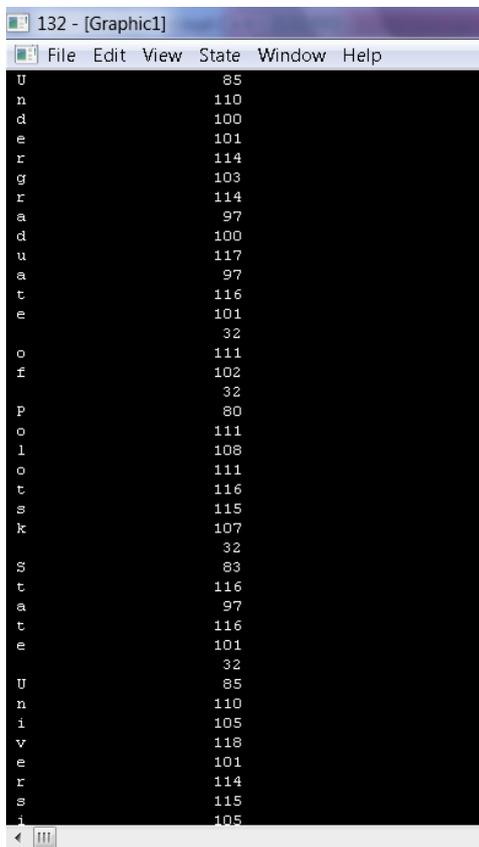


Рис.8

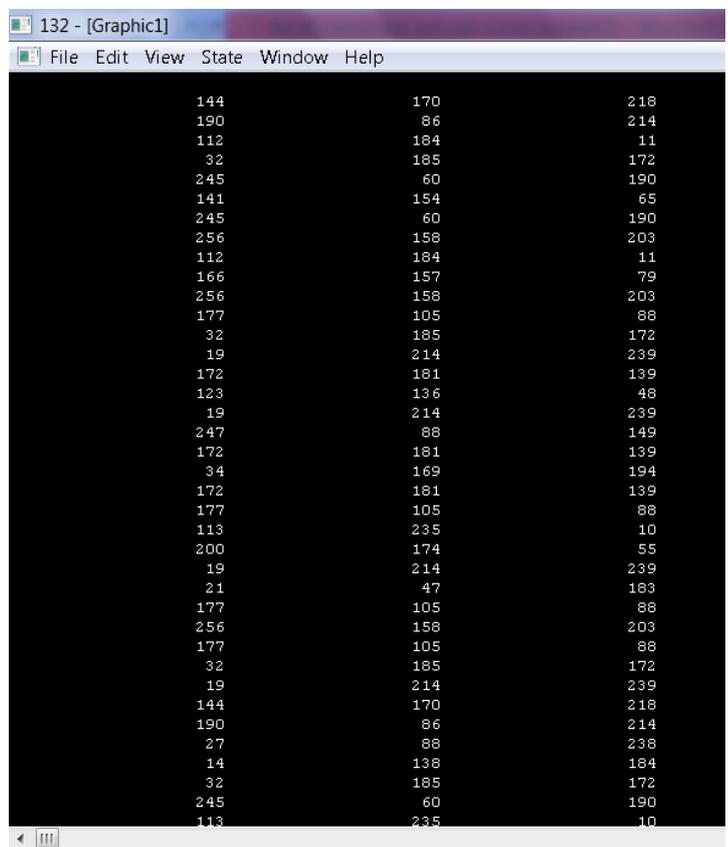


Рис.9

Исходная фраза для шифрования

$$(R(x, n), Q(x, y, n), Z(x, y, n) \pmod{p}), p = 257, y(x) = \exp(x), n = 8$$

На рисунке 9 видно, что шифр всех символов фразы s="Undergraduate of State University 2020"(магистранты Полоцкого государственного университета 2020 года) представляет тройки целых чисел из множества. То есть имеем отображение при $Z_p \rightarrow Z_p \times Z_p \times Z_p$ шифрования и $Z_p \times Z_p \times Z_p \rightarrow Z_p$ при дешифровании.

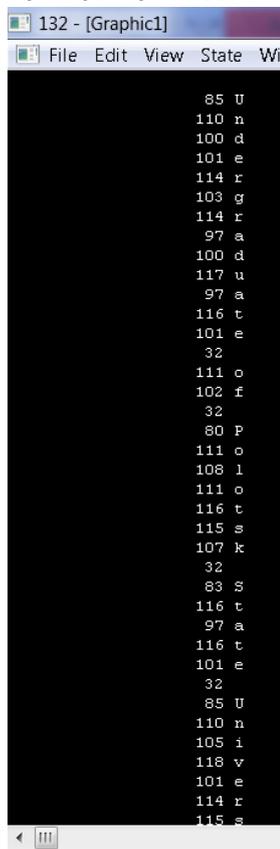


Рис.10

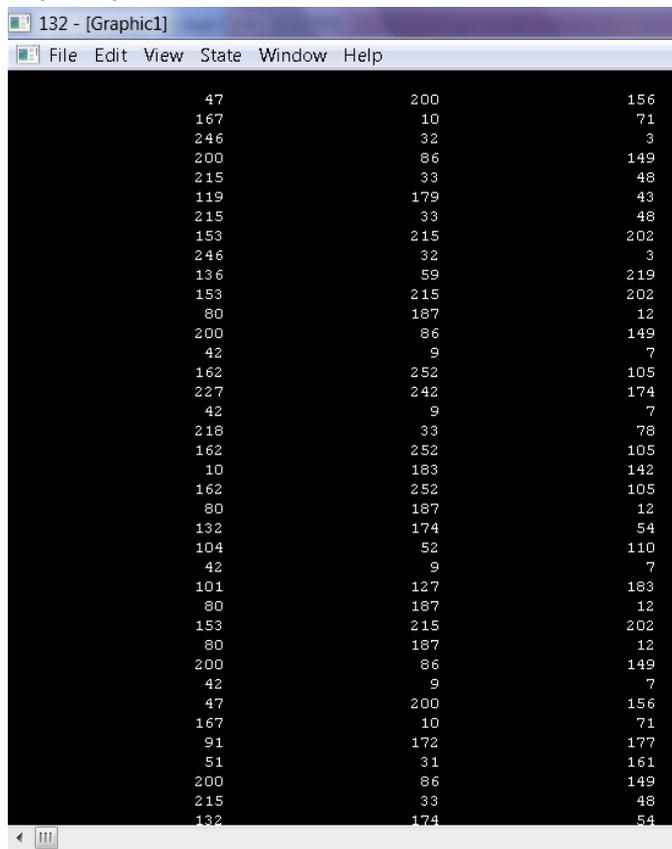


Рис.11

Результат дешифрования с параметрами $(R(x, n), Q(x, y, n), Z(x, y, n) \pmod{p}), p = 257, y(x) = \exp(x), n = 11$

$$R_{\exp(x)}^{n=8}, p = 257$$

Сравнивая рисунки 9,11 видно, что росток одной и той же фразы при шифровании одной и той же функцией, но с разным числом слагаемых в ростке имеет разные несовпадающие тройки чисел для всех символов одновременно! Поэтому параметр n-число слагаемых в ростке естественно использовать в качестве ключа.

Усложним алгоритм шифрования(29)-(35), введем дополнительный целый ключ a , и проще его ввести в третью часть шифра

$$Z(x, n, a) = \frac{ax^{n-1}}{n!} \pmod{p} = ax^{n-1} \cdot (n!)^{-1} \pmod{p}, (n!)^{-1} : (n!)^{-1} \cdot (n!) \equiv 1 \pmod{p}, 1 \leq a \leq p-1 \quad (36)$$

Тогда имеем формулы шифрования

$$(R(x, n), Q(x, n), Z(x, n) \pmod{p}) = \left(R_{y(x)}^{n-1}(x), R_{y(x)}^n(x), a \frac{x^{n-1}}{n!} \right) \pmod{p} \quad (37)$$

То есть

$$R(x, n) = R_{\exp(x)}^{n-1}(x) = \left(1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \dots + \frac{x^{n-1}}{n-1!} \right) \pmod{p} = \sum_{i=0}^{n-1} x^i \cdot (i!)^{-1} \pmod{p}, (i!)^{-1} : (i!)^{-1} \cdot (i!) \equiv 1 \pmod{p} \quad (38)$$

$$Q(x, n) = R_{\exp(x)}^{n-1}(x) = \left(1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \dots + \frac{x^n}{n!}\right) \pmod{p} = \sum_{i=0}^n x^i \cdot (i!)^{-1} \pmod{p}, (i!)^{-1} : (i!)^{-1} (i!) \equiv 1 \pmod{p} \quad (39)$$

Алгоритм дешифрования описывается формулой(40)

$$x \equiv (Q(x, n) - R(x, n))(Z(x, n))^{-1} \cdot a \pmod{p}, (Z(x, n))^{-1} : (Z(x, n))^{-1} \cdot Z(x, n) \equiv 1 \pmod{p}, a \neq 0 \pmod{p} \quad (40)$$

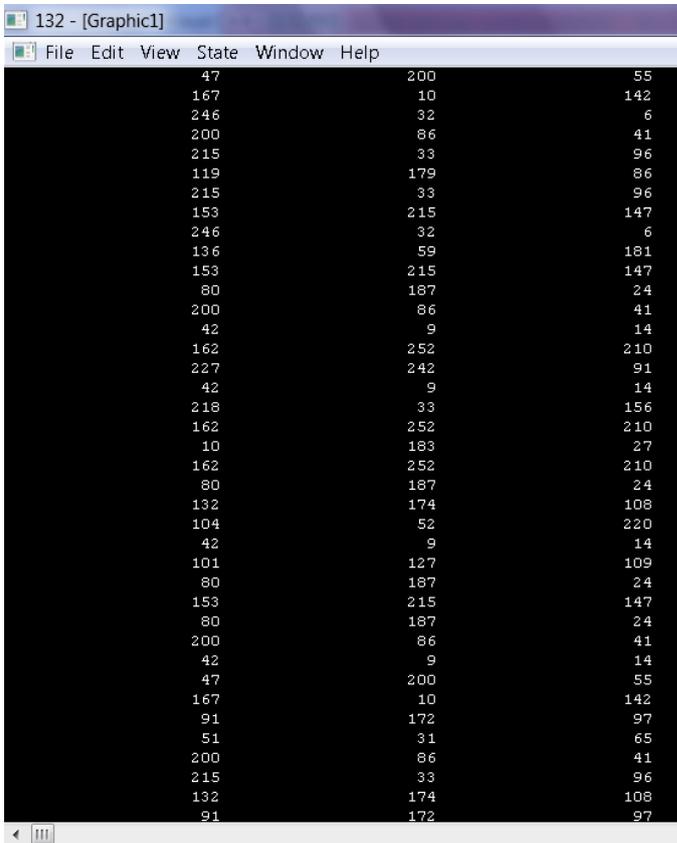


Рис.12

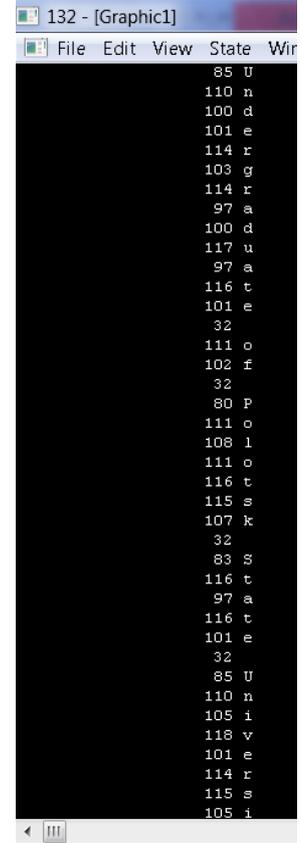


Рис.13

На рисунках 12,13 показан шифр модифицированного алгоритма (36)-(39) с параметрами $P=257, n=11, a=10, y(x)=\exp(x)$. На рисунке 13 видно, что дешифрование фразы Undergraduate of Polotsk State University 2020 происходит взаимно однозначно. Ниже написана программа модифицированного алгоритма(36)-(39)

```

program algebra
integer(8),parameter::n=11,p=257,len=46
integer(8)::x,y,z0,z1,z2,z3,z4,z00,mas1(len+1),mas2(len+1),mas3(len+1)
integer(8)::a,b,t,i,j,shifr(3,len+1),aa
character(len+2)::s
integer(8)::mas(len+2)
s="Undergraduate of Polotsk State University 2020"
do i=1,len
mas(i)=ichar(s(i:i))
print*,s(i:i),mas(i)
enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
pause
do j=1,len
x=mas(j)
a=1;aa=2
do i=1,n-1
call factor(x,i,p,z0)

```

```

a=mod(a+z0,p)
enddo
shifr(1,j)=a
b=1
do i=1,n
call factor(x,i,p,z0)
b=mod(b+z0,p)
enddo
shifr(2,j)=b
call factorial(x,n,p,z0)
shifr(3,j)=mod(z0*aa,p)
print*,shifr(1,j),shifr(2,j),shifr(3,j)
enddo
pause
do j=1,len
a=shifr(2,j)-shifr(1,j)
call f3(shifr(3,j),p,z0)
!call f3(aa,p,z1)
b=mod(z0*a*aa,p)
if(b<0)then
b=b+p
endif
print*,b,char(b)
enddo
pause
end program algebra
subroutine factor(x,i,p,z0)
integer(8)::x,i,p,k,z,z1,z0
z=1
do k=1,i
z=mod(z*x,p)
enddo
z1=1
do k=1,i
z1=mod(z1*k,p)
enddo
call f3(z1,p,z0)
z0=mod(z0*z,p)
end subroutine
subroutine factorial(x,i,p,z0)
integer(8)::x,i,p,k,z,z1,z0
z=1
do k=1,i-1
z=mod(z*x,p)
enddo
z1=1
do k=1,i
z1=mod(z1*k,p)
enddo
call f3(z1,p,z0)
z0=mod(z0*z,p)
end subroutine

```

```

subroutine f3(t,p,z0)
integer(8)::t,p,i,z0
do i=1,p-1
if(mod(i*t,p)==1)then
z0=i
endif
enddo
end subroutine

```

Рассмотрим теперь возможность целочисленного шифрования ростками гиперболических функций.

Пример 5(Algebra 5).

Обозначим росток элементарной функции $y(x) = ch(x) = \frac{\exp x + \exp(-x)}{2}$ с n слагаемыми как

$$R_{ch(x)}^n(x) = \frac{1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \dots + \frac{x^n}{n!} + 1 - x + \frac{x^2}{2} - \frac{x^3}{6} + \dots + (-1)^n \frac{x^n}{n!}}{2} = 1 + \frac{x^2}{2} + \frac{x^4}{4!} + \dots + \frac{x^{2k}}{(2k)!} \quad (41)$$

Пусть целое число $n = 2k$ и вид функции $y(x)$ ключи шифрования, x – номер текущего символа в сообщении по таблице ASCII(натуральное число $x \in N, 0 \leq x \leq p-1$). Введем обозначения

$$R(x, n = 2k) = R_{ch(x)}^{2k-2}(x) = 1 + x^2 + \frac{x^4}{4!} + \dots + \frac{x^{2(k-1)}}{(2(k-1))!}, Q(x, y, n = 2k) = R_{ch(x)}^n(x) = 1 + \frac{x^2}{2} + \frac{x^4}{4!} + \dots + \frac{x^{2k}}{(2k)!}, Z(x, n = 2k) = \frac{x^{2k-1}}{(2k)!} \quad (42)$$

В формуле(42) $R(x, n), Q(x, n), Z(x, n)$ определены при всех целых числах. Рассмотрим алгебраические преобразования(42) на множестве целых положительных остатков Z_p

Шифром (формула(31)) с ключами $n, p, n=2k$ символа x назовем тройку целых чисел $(R(x, 2k), Q(x, 2k), Z(x, 2k) \pmod p)$

$$(R(x, 2k), Q(x, 2k), Z(x, 2k) \pmod p) = \left(R_{y(x)}^{n-1}(x), R_{y(x)}^n(x), \frac{x^{2k-1}}{(2k)!} \right) \pmod p \quad (43)$$

То есть

$$R(x, 2k) = R_{ch(x)}^{2k-2}(x) = \left(1 + \frac{x^2}{2} + \frac{x^4}{4!} + \dots + \frac{x^{2(k-1)}}{(2(k-1))!} \right) \pmod p = \sum_{i=0}^{k-1} x^{2i} \cdot (2i!)^{-1} \pmod p, (2i!)^{-1} : (2i!)^{-1} (2i!) \equiv 1 \pmod p \quad (44)$$

$$Q(x, 2k) = R_{ch(x)}^{2k}(x) = \left(1 + x^2 + \frac{x^4}{4!} + \dots + \frac{x^{2k}}{(2k)!} \right) \pmod p = \sum_{i=0}^k x^{2i} \cdot (2i!)^{-1} \pmod p, (2i!)^{-1} : (2i!)^{-1} (2i!) \equiv 1 \pmod p \quad (45)$$

$$Z(x, 2k) = a \frac{x^{2k-1}}{(2k)!} \pmod p = ax^{2k-1} \cdot (2k!)^{-1} \pmod p, (2k!)^{-1} : (2k!)^{-1} (2k!) \equiv 1 \pmod p \quad (46)$$

Из определения шифра формулы(44),(45),(46), видно, что они корректно определены при любых целых значениях $x, n = 2k, 2k < p$

Дешифрование проводим по формуле(47)

$$x \equiv (Q(x, 2k) - R(x, 2k))(Z(x, 2k))^{-1} \cdot a \pmod p, (Z(x, 2k))^{-1} : (Z(x, 2k))^{-1} \cdot Z(x, 2k) \equiv 1 \pmod p \quad (47)$$

Из формул(46)следует, что для корректности необходимо условие $1 \leq x \leq p-1$. Далее следует программный код нашего алгоритма (41)-(47). Отметим, что мы сразу написали модифицированный алгоритм с параметром ключа $a=72$

```

program algebra;
integer(8),parameter::n=10,p=257,len=46,kk=n/2;
integer(8)::x,y,z0,z1,z2,z3,z4,z00,mas1(len+1),mas2(len+1),mas3(len+1);
integer(8)::a,b,t,i,j,shifr(3,len+1),aa;
character(len+2)::s;

```

```

integer(8)::mas(len+2);
s="Undergraduate of Polotsk State University 2020";
do i=1,len;
mas(i)=ichar(s(i:i));
print*,s(i:i),mas(i);
enddo; !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
pause;
do j=1,len;
x=mas(j);
a=1;aa=1;
do i=1,kk-1;
call factor(x,2*i,p,z0);
a=mod(a+z0,p);
enddo;
shifr(1,j)=a;
b=1;
do i=1,kk;
call factor(x,2*i,p,z0);
b=mod(b+z0,p);
enddo;
shifr(2,j)=b;
call factorial(x,2*kk,p,z0);
shifr(3,j)=mod(z0*aa,p);
print*,shifr(1,j),shifr(2,j),shifr(3,j);
enddo;
pause;
do j=1,len;
a=shifr(2,j)-shifr(1,j);
call f3(shifr(3,j),p,z0);
!call f3(aa,p,z1)
b=mod(z0*a*aa,p);
if(b<0)then;
b=b+p;
endif;
print*,b,char(b);
enddo;
pause;
end program algebra;
subroutine factor(x,i,p,z0);
integer(8)::x,i,p,k,z,z1,z0;
z=1;
do k=1,i;
z=mod(z*x,p);
enddo;
z1=1;
do k=1,i;
z1=mod(z1*k,p);
enddo;
call f3(z1,p,z0);
z0=mod(z0*z,p);
end subroutine;
subroutine factorial(x,i,p,z0);

```

```

integer(8)::x,i,p,k,z,z1,z0;
z=1;
do k=1,i-1;
z=mod(z*x,p);
enddo;
z1=1;
do k=1,i;
z1=mod(z1*k,p);
enddo;
call f3(z1,p,z0);
z0=mod(z0*z,p);
end subroutine;
subroutine f3(t,p,z0);
integer(8)::t,p,i,z0;
do i=1,p-1;
if(mod(i*t,p)==1)then;
z0=i;
endif;
enddo;
end subroutine;

```

```

U      85
n      110
d      100
e      101
r      114
g      103
r      114
a      97
d      100
u      117
a      97
t      116
e      101
      32
o      111
f      102
      32
p      80
o      111
l      108
o      111
t      116
s      115
k      107
      32
s      83
t      116
a      97
t      116
e      101
      32
U      85
n      110
i      105
v      118
e      101
r      114
s      115
i      105
t      116
y      121
      32
2      50
0      48
2      50
0      48
Fortran Pause - Enter command<CR>

```

Рис.14

```

133 - [Graphic1]
      215      132      226
      109      119      100
      67      100      106
      167      7      232
      228      242      90
      219      178      146
      228      242      90
      23      189      208
      67      100      106
      50      146      237
      23      189      208
      187      62      206
      167      7      232
      206      26      109
      216      86      131
      98      213      187
      206      26      109
      189      19      104
      216      86      131
      165      185      99
      216      86      131
      187      62      206
      240      63      173
      183      108      36
      206      26      109
      161      118      133
      187      62      206
      23      189      208
      187      62      206
      167      7      232
      206      26      109
      215      132      226
      109      119      100
      127      18      241
      240      212      192
      167      7      232
      228      242      90
      240      63      173
      127      18      241
      187      62      206
      212      106      41
      206      26      109

```

Рис.15

Результат шифрования с параметрами $(R(x,2k), Q(x,2k), Z(x,2k)(\text{mod } p)), y(x) = ch(x), n = 10, a = 72$

$$R_{ch(x)}^{n=10}, p = 257$$

Сравнивая рисунки 14,15 видно, что росток функции $y(x) = ch(x) = \frac{\exp(x) + \exp(-x)}{2}$ фразы="Undergraduate of Polotsk State University 2020" с числом слагаемых 10 в ростке $R_{ch(x)}^{n=10}$, $p = 257$ представляет тройки чисел для всех символов одновременно! Поэтому параметр n число слагаемых в ростке, а также дополнительный параметр $a=72$ естественно использовать в качестве ключей.

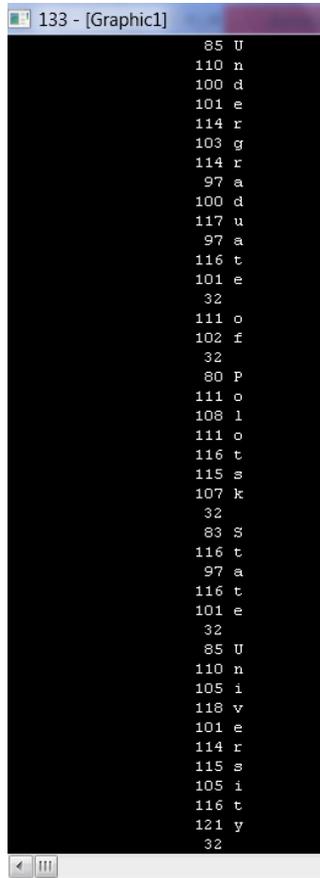


Рис.16 Результат дешифрования ростком гиперболического косинуса с параметрами $y(x) = ch(x)$, $n=10$, $a=72$.

Пример 6(Algebra 6).

Аналогично можно написать алгоритм шифрования – дешифрования для ростка гиперболического синуса.

Обозначим росток элементарной функции $y(x) = sh(x) = \frac{\exp(x) - \exp(-x)}{2}$ с n слагаемыми как

$$R_{ch(x)}^n(x) = \frac{1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \dots + \frac{x^{2k+1}}{2k+1!} - \left(1 - x + \frac{x^2}{2} - \frac{x^3}{6} + \dots + (-1)^{2k+1} \frac{x^{2k+1}}{2k+1!} \right)}{2} = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \dots + \frac{x^{2k+1}}{(2k+1)!} \quad (48)$$

Пусть целое число $n = 2k + 1$ и вид функции $y(x)$ ключи шифрования, x – номер текущего символа в сообщении по таблице ASCII(натуральное число $x \in N, 0 \leq x \leq p - 1$). Введем обозначения

$$R(x, n = 2k + 1) = R_{sh(x)}^{2k+1}(x) = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \dots + \frac{x^{2k+1}}{(2k+1)!}, Q(x, n = 2k + 1) = R_{sh(x)}^{2k+1}(x) = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \dots + \frac{x^{2k+1}}{(2k+1)!} \quad (49)$$

В формуле(49) $R(x,2k+1), Q(x,2k+1), Z(x,2k+1)$ определены при всех целых числах. Рассмотрим алгебраические преобразования(49) на множестве целых положительных остатков Z_p

Шифром (формула(50)) с ключами $p, n = 2k+1$ символа y назовем тройку целых чисел $(R(x,2k+1), Q(x,2k+1), Z(x,2k+1)) \pmod{p}$

$$(R(x,2k+1), Q(x,2k+1), Z(x,2k+1)) \pmod{p} = \left(R_{y(x)}^{2k-1}(x), R_{y(x)}^{2k+1}(x), \frac{x^{2k}}{(2k+1)!} \right) \pmod{p} \quad (50)$$

То есть

$$R(x,2k+1) = R_{sh(x)}^{2k-1}(x) = \left(x + \frac{x^3}{3!} + \frac{x^5}{5!} + \dots + \frac{x^{2k-1}}{(2k-1)!} \right) \pmod{p} = \sum_{i=0}^{k-1} x^{2i+1} \cdot (2i+1)!^{-1} \pmod{p}, (2i+1)!^{-1} : (2i+1)!^{-1} (2i+1)! \equiv 1 \pmod{p} \quad (51)$$

$$Q(x,2k+1) = R_{sh(x)}^{2k+1}(x) = \left(x + \frac{x^3}{3!} + \frac{x^5}{5!} + \dots + \frac{x^{2k+1}}{(2k+1)!} \right) \pmod{p} = \sum_{i=0}^k x^{2i+1} \cdot (2i+1)!^{-1} \pmod{p}, (2i+1)!^{-1} : (2i+1)!^{-1} (2i+1)! \equiv 1 \pmod{p} \quad (52)$$

$$Z(x,2k) = \frac{x^{2k}}{2k+1!} \pmod{p} = x^{2k} \cdot (2k+1)!^{-1} \pmod{p}, (2k+1)!^{-1} : (2k+1)!^{-1} \cdot (2k+1)! \equiv 1 \pmod{p} \quad (53)$$

Из определения шифра формулы(51),(52),(53), видно, что они корректно определены при любых целых значениях $x, n = 2k+1, 2k+1 < p$

Дешифрование проводим по формуле(54)

$$x \equiv (Q(x,2k+1) - R(x,2k+1)) (Z(x,2k))^{-1} \pmod{p}, (Z(x,2k))^{-1} : (Z(x,2k))^{-1} \cdot Z(x,2k) \equiv 1 \pmod{p} \quad (54)$$

Из формул(53) следует, что для корректности необходимо условие $1 \leq x \leq p-1$.

program algebra;

integer(8),parameter::n=10,p=257,len=46,kk=n/2;

integer(8)::x,y,z0,z1,z2,z3,z4,z00,mas1(len+1),mas2(len+1),mas3(len+1);

integer(8)::a,b,t,i,j,shifr(3,len+1),aa ;

character(len+2)::s;

integer(8)::mas(len+2);

s="Undergraduate of Polotsk State University 2020";

do i=1,len;

mas(i)=ichar(s(i:i)) ;

print*,s(i:i),mas(i);

enddo;!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

pause ;

do j=1,len ;

x=mas(j) ;

a=1;

aa=11 ;

do i=0,kk-1;

call factor(x,2*i+1,p,z0);

a=mod(a+z0,p);

enddo;

shifr(1,j)=a;

b=1;

do i=0,kk;

call factor(x,2*i+1,p,z0);

b=mod(b+z0,p) ;

enddo;

shifr(2,j)=b;

call factorial(x,2*kk+1,p,z0) ;

shifr(3,j)=mod(z0*aa,p);

print*,shifr(1,j),shifr(2,j),shifr(3,j);

```

enddo;
  pause;
do j=1,len;
a=shifr(2,j)-shifr(1,j) ;
call f3(shifr(3,j),p,z0);
b=mod(z0*a*aa,p);
if(b<0)then;
b=b+p;
endif;
  print*,b,char(b);
enddo;
  pause ;
end program algebra;
subroutine factor(x,i,p,z0);
  integer(8)::x,i,p,k,z,z1,z0;
  z=1 ;
do k=1,i;
  z=mod(z*x,p);
enddo ;
z1=1;
do k=1,i ;
z1=mod(z1*k,p);
enddo ;
call f3(z1,p,z0);
z0=mod(z0*z,p);
end subroutine;
subroutine factorial(x,i,p,z0);
  integer(8)::x,i,p,k,z,z1,z0;
  z=1 ;
do k=1,i - 1 ;
z=mod(z*x,p);
enddo ;
z1=1 ;
do k=1,i;
z1=mod(z1*k,p);
enddo;
call f3(z1,p,z0);
z0=mod(z0*z,p);
end subroutine ;
subroutine f3(t,p,z0);
  integer(8)::t,p,i,z0 ;
do i=1,p - 1;
  if(mod(i*t,p)==1)then;
z0=i ;
  endif;
enddo;
end subroutine;

```

Пример 7(Algebra 7).

В данном разделе мы опишем идею шифрования текста квадратными матрицами-ключами с элементами из поля $Z_p = \{0,1,2,\dots, p-1\}$, а также алгоритм шифрования на классе правых квазипрямоугольных матриц. Исходную фразу из k символов клавиатуры располагаем в одномерный массив $mas[i], i=\overline{1,k}$. Далее, исходную фразу из k элементов нужно расположить в прямоугольную матрицу $U(m \times n)$. Если $k \neq m \cdot n$, то $m = \left\lceil \frac{k}{n} \right\rceil + 1$ - число строк в текстовой матрице, а n – число ее столбцов. (55)

На рисунках 17,18 показано размещение текста в одномерный массив и в матрицу размером 5×6 . Пусть квадратная матрица-ключ $A(m \times m)$ не содержит нулевых элементов на ее главной диагонали, это требование необходимо для упрощения алгоритма обращения матрицы-ключа.

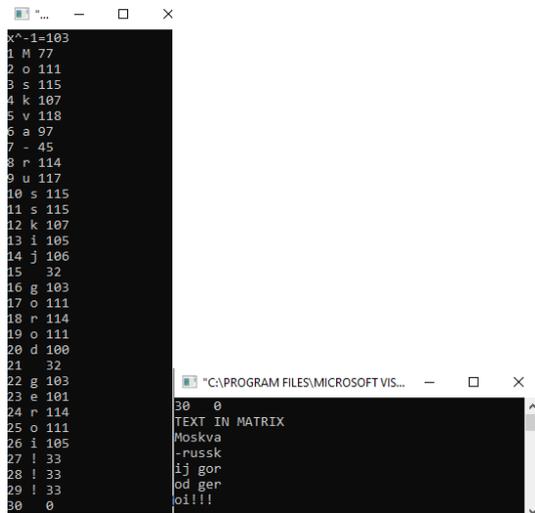


Рис.17. Исходная фраза текста "Moskva-russkij gorod geroi!!!" для шифрования записана в одномерном массиве.

Рис.18. Исходная фраза текста "Moskva-russkij gorod geroi!!!" для шифрования записана в матрице $U(5 \times 6)$.

Матричное шифрование проводим по формуле(56)

$$A(m \times m) \cdot U(m \times n) \equiv S(m \times n) \pmod{p} \tag{56}$$

В формуле(56) $A(m \times m)$ -матрица-ключ, $U(m \times n)$ - текстовая матрица, $S(m \times n)$ - матрица-шифр. Поэлементная запись матрицы S показана с помощью формулы(57)-первый шаг

$$1. s_{i,j} \equiv \sum_{l=1}^m a_{i,l} u_{l,j} \pmod{p}, i = \overline{1,m}, j = \overline{1,n} \tag{57}$$

Для дешифрования нужно обратить матрицу A . Используем для этого блочный алгоритм. Слева расположим матрицу A , Справа единичную матрицу той же размерности. Затем элементарными операциями над строками привести \bar{A} к такой матрице, у которой слева расположена единичная матрица, тогда справа будет обратная A^{-1} к исходной матрице-ключу A^{-1} по простому модулю p . Кратко алгоритм описывается формулой(58)

$$\bar{A}(m \times 2m) = (A(m \times m) | E(m \times m)) \sim (E(m \times m) | A^{-1}(m \times m)) \pmod{p} \tag{58}$$

Более подробно алгоритм в формуле(58) можно записать в виде(59) A^{-1} : (второй и третий шаг алгоритма)

$$2. k = \overline{1,m}: a_{k,j}^{new} \equiv a_{k,j}^{old} \cdot (a_{k,k}^{old})^{-1} \pmod{p}, j = \overline{1,2m}, (a_{k,k}^{old})^{-1} := (a_{k,k}^{old})^{-1} \cdot a_{k,k}^{old} \equiv 1 \pmod{p} \tag{59}$$

$$3. a_{i,j}^{new} \equiv a_{i,j}^{old} - a_{i,k}^{old} \cdot a_{k,j}^{old} \pmod{p}, j = \overline{1,2m}, i = \overline{1,k-1} \cup \overline{k+1,m} \tag{59}$$

$$4. a_{i,j}^{-1} = a_{i,j-m}^{new}, j = \overline{m+1,2m}, i = \overline{1,m} \tag{60}$$

После четвертого шага алгоритма, по формуле(60) получаем обратную матрицу $A^{-1} \pmod{p}$. Дешифрование проводим по формуле(61) в матричном виде

$$A \cdot U \equiv S \pmod{p} \Leftrightarrow U \equiv A^{-1} \pmod{p} \cdot S \pmod{p} \quad (61)$$

Или подробнее для каждого элемента дешифрованной матрицы получим(62)- пятый шаг алгоритма

$$5. u_{i,j} \equiv \sum_{l=1}^m a^{-1}_{i,l} S_{l,j} \pmod{p}, i = \overline{1,m}, j = \overline{1,n} \quad (62)$$

Из второго шага алгоритма, формула(59), видно, что для корректности всего алгоритма достаточно, чтобы диагональные элементы матрицы-ключа A имели обратные элементы, то есть достаточно, чтобы все диагональные элементы матрицы A принадлежали расширенной группе $Z_p / 0 = \{1, 2, \dots, p-1\}$.

Каждый символ текстовой матрицы U или матрицы шифра S можно распечатать в символьном виде (клавиатуры ASCII) или в целочисленном виде. Для тестирования алгоритма(56)-(62) и соответствующей алгоритму программы мы выбрали ключ-матрицу вида с простым модулем $p=257$.

$$A = \begin{pmatrix} 4 & 2 & 2 & 5 & 1 \\ 3 & 3 & 1 & 4 & 3 \\ 2 & 5 & 2 & 1 & 2 \\ 2 & 1 & 4 & 4 & 2 \\ 1 & 4 & 2 & 2 & 5 \end{pmatrix} \quad (63)$$

Найдем с помощью программы матрицу обратную по(mod p) к исходной A , которая представлена на Рис.20.

На Рис.21 программой выполнена проверка $A^{-1} \cdot A \equiv I \pmod{p}$. На этом же рисунке по формулам (61),(62) произведено дешифрование. Сравнивая рисунки 18,21 убеждаемся, что исходный текст фразы "Moskva-russkij gorod geroi!!!" посимвольно совпадает с дешифрованным текстом. Таким образом, алгоритм(56)-(62) и соответствующая алгоритму программа шифруют устойчиво.

Отметим также особенности программы. Матрицу ключ(63) нельзя набирать с помощью формул, иначе программа останавливается с сообщением "Tramp". То есть, каждый элемент матрицы приходится набирать отдельно. Далее, поиск обратного числа $(a_{k,k}^{old})^{-1} \pmod{p}$ к числу $a_{k,k}^{old}$ из множества остатков $\{1, 2, \dots, p-1\}$ выполняет отдельная подпрограмма `int obr(int x, int p)`.

Из - за достаточно большого кода программы, необходимо провести некоторые комментарии внутри программы, что и было сделано нами.

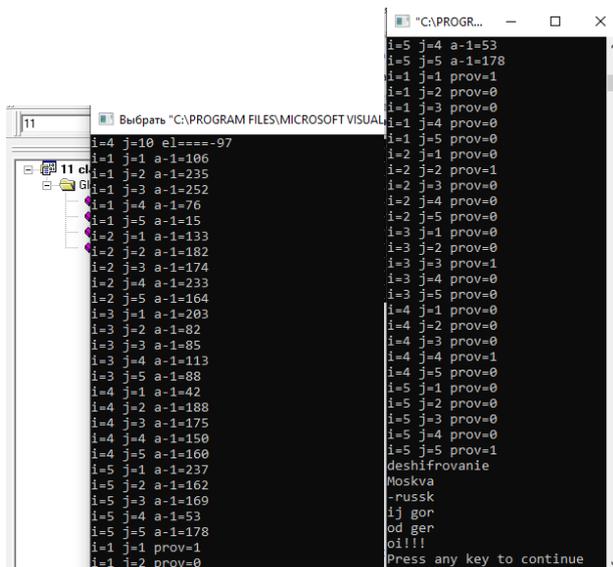


Рис.20. Для исходной матрицы(63) найдена обратная матрица $A^{-1}(\text{mod}257)$.

Рис.21. Проверка условия $A^{-1} \cdot A \equiv 1(\text{mod}p)$ программой, а также дешифрованный текст. Далее нами написана программа с комментариями, реализующая алгоритм(56)-(62).

```

#include <stdio.h>
#include <math.h>
int const k = 30, m = 5, n = 6, p = 257;
int obr(int x, int p)
{
    int s, i;
    s = 1;
    for (i = 1; i <= p - 1; i++)
    {
        s = (i * x) % p;
        if (s < 0)
        {
            s = s + p;
        }
        if (s == 1)
        {
            return i;
        }
    }
}
int main()
{
    int s, s1, ss, i, j, l, jj, kk, matr00[m + 1][2 * m + 1], mas[k + 1], matr[m + 1][n + 1],
    matr0[m + 1][2 * m + 1], matr2[m + 1][m + 1], matr3[m + 1][n + 1], matr21[m + 1][m + 1],
    matr4[m + 1][n + 1], matr33[m + 1][m + 1];
    char str[k + 1] = "Moskva-russkij gorod geroi!!!";
    printf("x^-1=%ld\n", obr(5, p));
    matr2[1][1] = 4; matr2[1][2] = 2; matr2[1][3] = 2;
    matr2[1][4] = 5; matr2[1][5] = 1 ;
}

```

```

matr2[2][1] = 3;matr2[2][2] = 3;matr2[2][3] = 1;
matr2[2][4] = 4;matr2[2][5] = 3;
matr2[3][1] = 2;matr2[3][2] = 5;matr2[3][3] = 2;
matr2[3][4] = 1;matr2[3][5] = 2;
matr2[4][1] = 2;matr2[4][2] = 1;matr2[4][3] = 4;
matr2[4][4] = 4;matr2[4][5] = 2;
matr2[5][1] = 1;matr2[5][2] = 4;matr2[5][3] = 2;
matr2[5][4] = 2;matr2[5][5] = 5;
//zadanie klucha;
for (i = 1; i <= k ; i++)
{
    mas[i] = str[i-1];
    printf("%d %c %d\n", i, mas[i], mas[i]);
}
printf("TEXT IN MATRIX\n");
for (s = 1; s <= k ; s++)
{
    jj=int(s/n);
    if(s%n==0)
    {
        j=n;
        i=jj;
    }
    else
    {
        j=s%n;
        i=(s-j)/n+1;
    }
    matr[i][j] = mas[s];
}
for(i=1;i<=m;i++)
{
    for(j=1;j<=n;j++)
    {
        printf("%c",matr[i][j]);
    }
    printf("\n");
}
//matrix multiply//////////
for (i = 1; i <= m; i++)
{
    for (j = 1; j <= n; j++)
    {
        ss = 0;
        for (l = 1; l <= m; l++)
        {
            ss = (ss + matr2[i][l] * matr[l][j]) % p;
            if (ss < 0)
            {
                ss = ss + p;
            }
        }
    }
}

```

```

        matr3[i][j] = ss;
    }
}
// matr3[i][j] shifr;
printf("shifr in column\n");
// text of matrix
for (i = 1; i <= m; i++)
{
    for (j = 1; j <= n; j++)
    {
        printf("%d %d %c\n",i,j,matr3[i][j]);
        //printf("%c",matr3[i][j]);
    }
    //printf("\n");
}

for (i = 1; i <= m; i++)
{
    for (j = 1; j <= 2 * m; j++)
    {
        if (j <= m)
        {
            matr0[i][j] = matr2[i][j];
        }
        else if (i == j - m && j > m)
        {
            matr0[i][j] = 1;
        }
        else if (j > m && !(i == j - m))
        {
            matr0[i][j] = 0;
        }
        else
        {
            matr0[i][j] = 0;
        }
        printf("%d %d A=%d\n", i, j, matr0[i][j]);
    }
}
//inverse matrix
for (kk = 1; kk <= m; kk++)
{
    s1= obr(matr0[kk][kk], p);
    printf("kk=%d s1=%d\n",kk,s1);
    for(j=1;j<=2*m;j++)
    {
        matr00[kk][j] = (matr0[kk][j] * s1)%p;
        printf("kk=%d j=%d el===== %d\n", kk, j, matr00[kk][j]);
    }
    for(j=1;j<=2*m;j++)
{
    matr0[kk][j] = matr00[kk][j] ;
}
}
}

```

```

    }
    for (i = 1; i <= m; i++)
    {
        if (!(i == kk))
        {
            for(j= 1;j <= 2 * m;j++)
            {
                matr00[i][j] =(matr0[i][j] - matr0[kk][j] * matr0[i][kk])%p;
                printf("i=%d j=%d el=====d\n", i, j, matr00[i][j]);
            }
        }
    }
    for(i=1;i<=m;i++)
    for(j=1;j<=2*m;j++)
    {
        matr0[i][j]=matr00[i][j];
        if(matr0[i][j]<0)
        {
            matr0[i][j]=matr0[i][j]+p;
        }
    }
}
//end kk
for (i = 1; i <= m; i++)
{
    for (j = m+1; j <= 2*m; j++)
    {
        matr21[i][j-m]=matr0[i][j];
        printf("i=%d j=%d a-1=%d\n",i,j-m,matr21[i][j-m]);
    }
}
// matr21[i][j])- inverse matrix;
//proverka inverse matrix;
for (i = 1; i <= m; i++)
{
    for (j = 1; j <= m; j++)
    {
        ss = 0;
        for (l = 1; l <= m; l++)
        {
            ss = (ss + matr2[i][l] * matr21[l][j]) % p;
            if (ss < 0)
            {
                ss = ss + p;
            }
        }
        matr33[i][j] = ss;
        printf("i=%d j=%d prov=%d\n",i,j,matr33[i][j]);
    }
}

printf("deshifrovanie\n");

```

```

//matrix multiply//////////
for (i = 1; i <= m; i++)
{
    for (j = 1; j <= n; j++)
    {
        ss = 0;
        for (l = 1; l <= m; l++)
        {
            ss = (ss + matr21[i][l] * matr3[l][j]) % p;
            if (ss < 0)
            {
                ss = ss + p;
            }
        }
        matr4[i][j] = ss;
        printf("%c",matr4[i][j] );
    }
    printf("\n");
}
}

```

Оценим мощность пространства ключей в примере 7. Число малых и больших букв английского языка $26+26=52$. Число арабских цифр 10. Добавим сюда разделительные знаки, пробел, тире, точка, запятая, точка с запятой. Итого, минимальное число символов 65. Так как кодирующая матрица состоит из 25 элементов, каждый из элементов принимает независимо любой из 65 возможных символов, то мощность пространства ключей равна $65^{25}=2 \cdot 10^{45}$. Если компьютер успевает перебирать матрицы-ключи под перехваченный шифр со скоростью 1 миллиард в секунду, то время перебора составит $7 \cdot 10^{28}$ лет. Даже перебор с помощью квантового компьютера такой мощности ключей практически невозможен.

Пример 8(Algebra 8).

В данном примере мы рассмотрим линейное пространство квазипрямоугольных матриц.

Определение. Квазипрямоугольной матрицей назовем матрицу с неполной нижней строкой. Соответственно правой квазипрямоугольной матрицей – матрицу, у которой заполнение нижней строки начинается с правого края. У левой квазипрямоугольной матрицы нижняя строка заполняется с левого края.

Левая квазипрямоугольная матрица возникает естественным образом при заполнении текстового файла произвольным символьным текстом, начиная с верхнего левого угла. Естественно рассматривать все вычисления над квазипрямоугольными матрицами в группе целочисленных остатков $Z_p = \{0,1,2,\dots,p-1\}$. Как и все вычисления, которые рассматривались в предыдущих 7 примерах. Для такой матрицы ее элементы имеют значения $a_{i,j} = a_{i,j} \pmod p, (i = \overline{0, m-1}, j = \overline{0, n}) \cup (i = m, j = \overline{0, j_0})$, где j_0 – номер столбца последнего символа в нижней строке. Легко превратить левую квазипрямоугольную матрицу в правую квазипрямоугольную матрицу по формуле $a_{i,j}^{right} = a_{i,n-j}^{left} \pmod p, (i = \overline{0, m-1}, j = \overline{0, n}) \cup (i = m, j = \overline{0, j_0})$

Пространство левых (правых) квазипрямоугольных матриц с неизменными порядками (m,n,j_0) образует линейное пространство L . Действительно, сумма двух левых квазипрямоугольных матриц является левой квазипрямоугольной матрицей из пространства L (обозначим сумму матриц $C=A+B \pmod p$). Если $A, B \in L | a_{i,j} = a_{i,j} \pmod p, b_{i,j} = b_{i,j} \pmod p, (i = \overline{0, m-1}, j = \overline{0, n}) \cup (i = m, j = \overline{0, j_0})$, то

$$c_{i,j} = (a_{i,j} + b_{i,j}) \pmod p, (i = \overline{0, m-1}, j = \overline{0, n}) \cup (i = m, j = \overline{0, j_0}), C = A + B \in L \quad (64)$$

Умножение квазипрямоугольной матрицы A на целое число λ из группы Z_p назовем квазипрямоугольную матрицу, вычисленную по формуле

$$b_{i,j} = a_{i,j} \lambda \pmod{p}, (i = \overline{0, m-1}, j = \overline{0, n}) \cup (i = m, j = \overline{0, j_0}), b_{i,j}, a_{i,j}, \lambda \in L$$

Приведенные формулы - определения показывают линейность пространства левых (правых) квазипрямоугольных матриц. Нулевым элементом линейного пространства служит нулевая квазипрямоугольная матрица с теми же заполненными позициями нулями, что и остальные матрицы пространства.

Идея шифрования текста из текстового файла, который представляет собой левую квазипрямоугольную матрицу с матричными элементами из поля $Z_p = \{0, 1, 2, \dots, p-1\}$ заключается в следующей формуле

$$\begin{cases} c_{i,j} = (a_{i,j} + \lambda b_{i,j}) \pmod{p}, (i = \overline{0, m-1}, j = \overline{0, n}) \cup (i = m, j = \overline{0, j_0}), b_{i,j}, a_{i,j}, \lambda \in L \\ \lambda = T_{j_0}(x) \pmod{p}, T_k(x) = (2xT_{k-1}(x) - T_{k-2}(x)) \pmod{p}, T_0(x) \equiv 1, T_1(x) \equiv x \pmod{p}, x \in Z_p, k \geq 2, \end{cases} \quad (65)$$

В формуле(65) число λ представляет собой целое число-ключ над полем остатков из группы Z_p . В свою очередь результирующий ключ λ является функцией - многочленом Чебышева первого рода двух ключей x, j_0 . Аргумент многочлена x - известный (заданный) ключ из группы остатков Z_p , j_0 -неизвестный ключ, - это номер столбца последнего символа-метки в нижней строке левой квазипрямоугольной матрицы текста $a_{i,j}$. Одновременно индекс j_0 является порядком многочлена Чебышева первого рода. Шифром в формуле(65) является левая квазипрямоугольная матрица $c_{i,j}$.

Дешифрование производится по обратной формуле

$$\begin{cases} a_{i,j} = (c_{i,j} - \lambda b_{i,j}) \pmod{p}, (i = \overline{0, m-1}, j = \overline{0, n}) \cup (i = m, j = \overline{0, j_0}), b_{i,j}, a_{i,j}, \lambda \in L \\ \lambda = T_{j_0}(x) \pmod{p}, T_k(x) = (2xT_{k-1}(x) - T_{k-2}(x)) \pmod{p}, T_0(x) \equiv 1, T_1(x) \equiv x \pmod{p}, x \in Z_p, k \geq 2, \end{cases} \quad (66)$$

Отметим также, что матрица $b_{i,j}$ в формулах(65),(66) представляет собой матрицу-шифр большой мощности $(Z_p)^M$, где число символов данной матрицы совпадает с числом символов исходного текста M . В данном примере мы использовали простую формулу для матрицы-шифра

$$b_{i,j} = i + j \pmod{p}, (i = \overline{0, m-1}, j = \overline{0, n}) \cup (i = m, j = \overline{0, j_0}) \quad (67)$$

Для создания программы данного примера 8 мы использовали компилятор Cxx для андроидного устройства. Андроидный вариант языка C++ (компилятор Cxx) прост в использовании, имеет минимальное число библиотек для поддержки текста любого языка (русский, английский и др.), открыт в доступе через Play Market, и загружается на андроидное устройство - телефон. Так как телефоны имеют почти все студенты, то вся группа студентов имеет возможность одновременно программировать независимо от того имеет ли данная аудитория оборудование с компьютерами или нет. Отметим также, что во всех формулах(65)-(67) использовался модуль группы остатков Z_p $p=256$. Именно такое число символов имеет клавиатура ASCII, при этом интерпретация любого символа, а также невидимых 33-х знаков табуляции с порядковыми номерами от 0 до 32(пробел) происходит взаимно-однозначно для соответствия символ клавиатуры - порядковый номер символа в ASCII.

Параллельно с шифрованием и дешифрованием текста программа создает текстовый файл 1.txt, куда записывается исходный текст в том порядке, в каком текст расположен в символьной строке программы с учетом знаков табуляции. В конце символьной строки программы стоит метка - число 0(порядковый номер нуля-48 в ASCII). Метка в конце текста для ускорения и уменьшения операций в работе программы при записи текста и его считывании из текстового файла *.txt. При этом шифр содержит все символы текста, включая и метку. Шифр исходного текста записывается в созданный программой файл 2.txt. Затем программа закрывает файл 2.txt, чтобы очистить данные о шифре из оперативной памяти компьютера (телефона). Затем файл 2.txt с шифром вновь

открывается, шифр считывается, дешифруется и полученные данные до метки (число 0) записываются на консоль и в текстовый файл 3.txt.

На Рис. 23 изображен вывод программы текста на русском языке на консоль. Сначала программа находит номер столбца с последним символом-меткой $j_0=16$. Также программа выводит во вторую строку значение многочлена Чебышева первого рода $T_{j_0}(x)=1(\text{mod}256)$, $x=13$, $j_0=16$. Это число $\lambda=T_{j_0}(x)$.

На Рис. 24. Показан текстовый файл 1.txt с текстом на русском и английском языках. Как видно из Рис.24 в конце текста последним символом записывается символ -метка 0.

На Рис.25 представлен шифр текста для русской и английской клавиатуры при заданных и ранее указанных ключах $x=13$, $j_0=16$. Как видно из рисунка 25 текст на русском языке шифруется менее плотно, чем английский из-за того, что часто присутствуют знаки разрыва столбца и разрыва страницы.

На Рис. 26 приведен окончательно прочитанный программой текст на русском языке (начало файла 3.txt) и конец текста на английском языке (конец файла 3.txt). Как видно из рисунков 23-26 шифрование и дешифрование текстовых файлов на русском и английском языках происходит без искажений. Сравнивая также Рис. 24 и Рис. 26 видим, что число их символов различается на один (числа символов указаны в левом нижнем угле рисунков 24 и 26) соответственно 1682 и 1681. Так как Рис. 26 уже не содержит последний неинформативный символ-метку символ"0".



16

1

Мороз и солнце;
день чудесный!
Еще ты дремлешь, друг прелестный –
Пора, красавица, проснись:
Открой сомкнуты негой взоры
Навстречу северной Авроры,
Звездою севера явись!

Вечор, ты помнишь, вьюга злилась,
На мутном небе мгла носилась;
Луна, как бледное пятно,
Сквозь тучи мрачные желтела,
И ты печальная сидела –
А нынче... погляди в окно:

Под голубыми небесами
Великолепными коврами,
Блестя на солнце, снег лежит;
Прозрачный лес один чернеет,
И ель сквозь иней зеленеет,
И речка подо льдом блестит.

Вся комната янтарным блеском
Озарена. Веселым треском
Трещит затопленная печь.
Приятно думать у лежанки.
Но знаешь: не велеть ли в санки
Кобылку бурую запречь?

Скользя по утреннему снегу,
Друг милый, предадимся бегу
Нетерпеливого коня
И навестим поля пустые,
Леса, недавно столь густые,
И берег, милый для меня.

Перевод

Frost and sun;
wonderful day!

Рис.23. Запись текста на русском языке на консольную часть

<p>☰ 1</p> <p>Нач.стр. Вставить Вид</p>	<p>☰ 1</p> <p>Нач.стр. Вставить Вид Про</p>
<p>Мороз и солнце; день чудесный! Еще ты дремлешь, друг прелестный – Пора, красавица, проснись: Открой сомкнуты негой взоры Навстречу северной Авроры, Звездой севера явись!</p> <p>Вечор, ты помнишь, вьюга злилась, На мутном небе мгла носилась; Луна, как бледное пятно, Сквозь тучи мрачные желтела, И ты печальная сидела – А нынче... погляди в окно:</p>	<p>Вся комната янтарным блеском Озарена. Веселым треском Трещит затопленная печь. Приятно думать у лежанки. Но знаешь: не велеть ли в санки Кобылку бурую запретить?</p> <p>Скользя по утреннему снегу, Друг милый, предадимся бегу Нетерпеливого коня И навестим поля пустые, Леса, недавно столь густые, И берег, милый для меня.</p> <p>Перевод</p> <p>Frost and sun; wonderful day! You are still dozing, lovely friend - It's time, beauty, wake up: Open eyes closed by bliss Towards the northern Aurora, Be the star of the north!</p>
<p>Под голубыми небесами Великолепными коврами, Блестя на солнце, снег лежит; Прозрачный лес один чернеет, И ель сквозь иней зеленеет, И речка подо льдом блестит.</p> <p>Вся комната янтарным блеском Озарена. Веселым треском Трещит затопленная печь. Приятно думать у лежанки. Но знаешь: не велеть ли в санки Кобылку бурую запретить?</p> <p>Скользя по утреннему снегу, Друг милый, предадимся бегу Нетерпеливого коня И навестим поля пустые, Леса, недавно столь густые, И берег, милый для меня.</p>	<p>Evening, do you remember, the blizzard was angry, In the cloudy sky there was a haze; The moon is like a pale spot Turned yellow through the gloomy clouds, And you sat sad - And now... look out the window:</p> <p>Under blue skies splendid carpets, Shining in the sun, the snow lies; The transparent forest alone turns black, And the spruce turns green through the hoarfrost, And the river glitters under the ice.</p>
<p>Перевод</p> <p>Frost and sun; wonderful day! You are still dozing, lovely friend - It's time, beauty, wake up: Open eyes closed by bliss Towards the northern Aurora, Be the star of the north!</p>	<p>The whole room amber gleam Enlightened. Cheerful crackling The fired oven crackles. It's nice to think by the couch. But you know: do not order to the sled Ban the brown filly?</p> <p>Gliding through the morning snow Dear friend, let's run impatient horse And let's visit the empty fields, The forests, recently so dense, And the shore, dear to me.0</p>
<p>Evening, do you remember, the blizzard was an In the cloudy sky there was a haze; Turned yellow through the gloomy clouds,</p> <p>Полный текст: 1682</p>	<p>Полный текст: 1682</p>

Рис. 24. Запись текста на русском и английском языках в текстовый файл 1.txt.

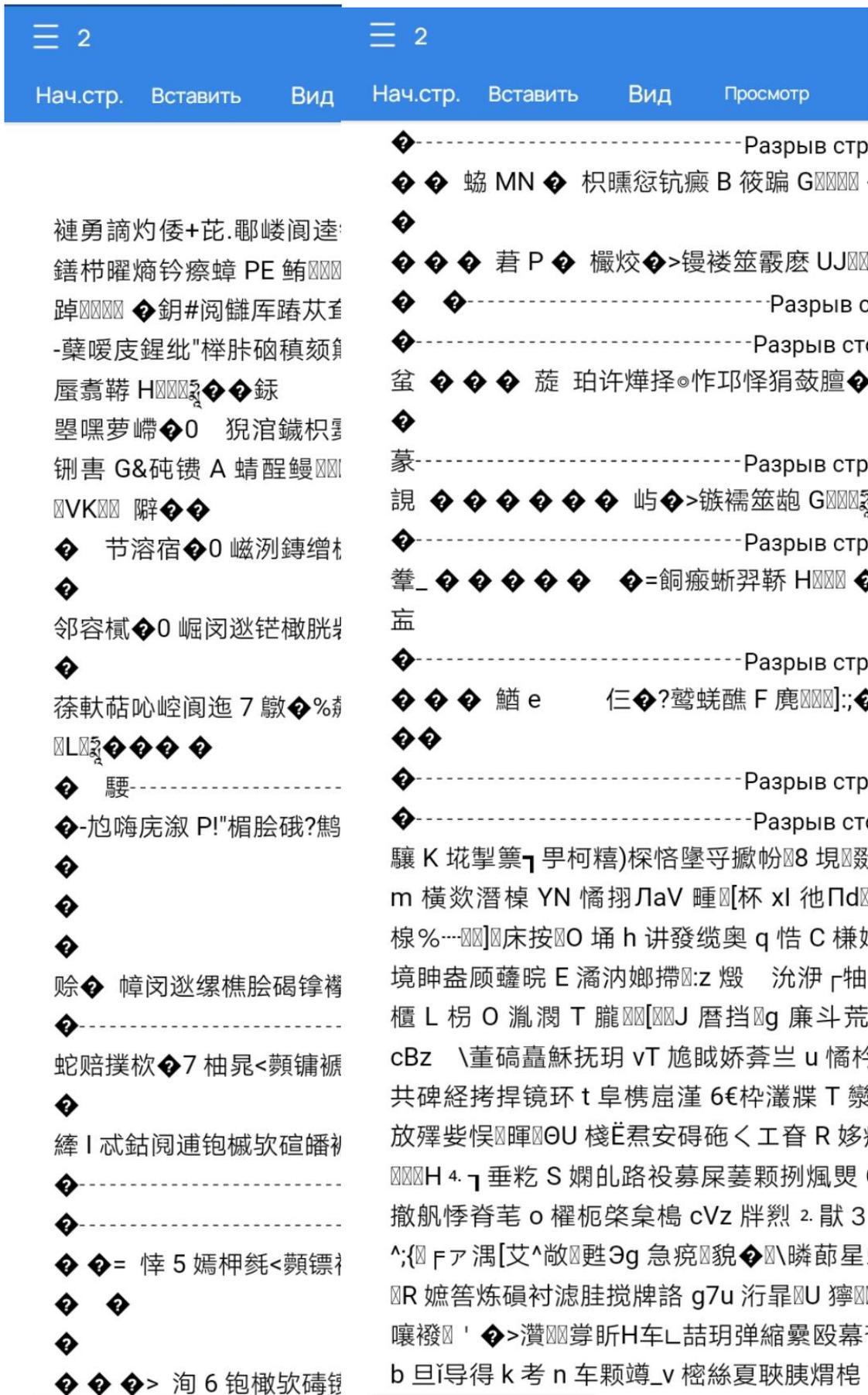


Рис. 25. Запись шифра русского (начало) и английского (конец) текста в текстовый файл 2.txt.

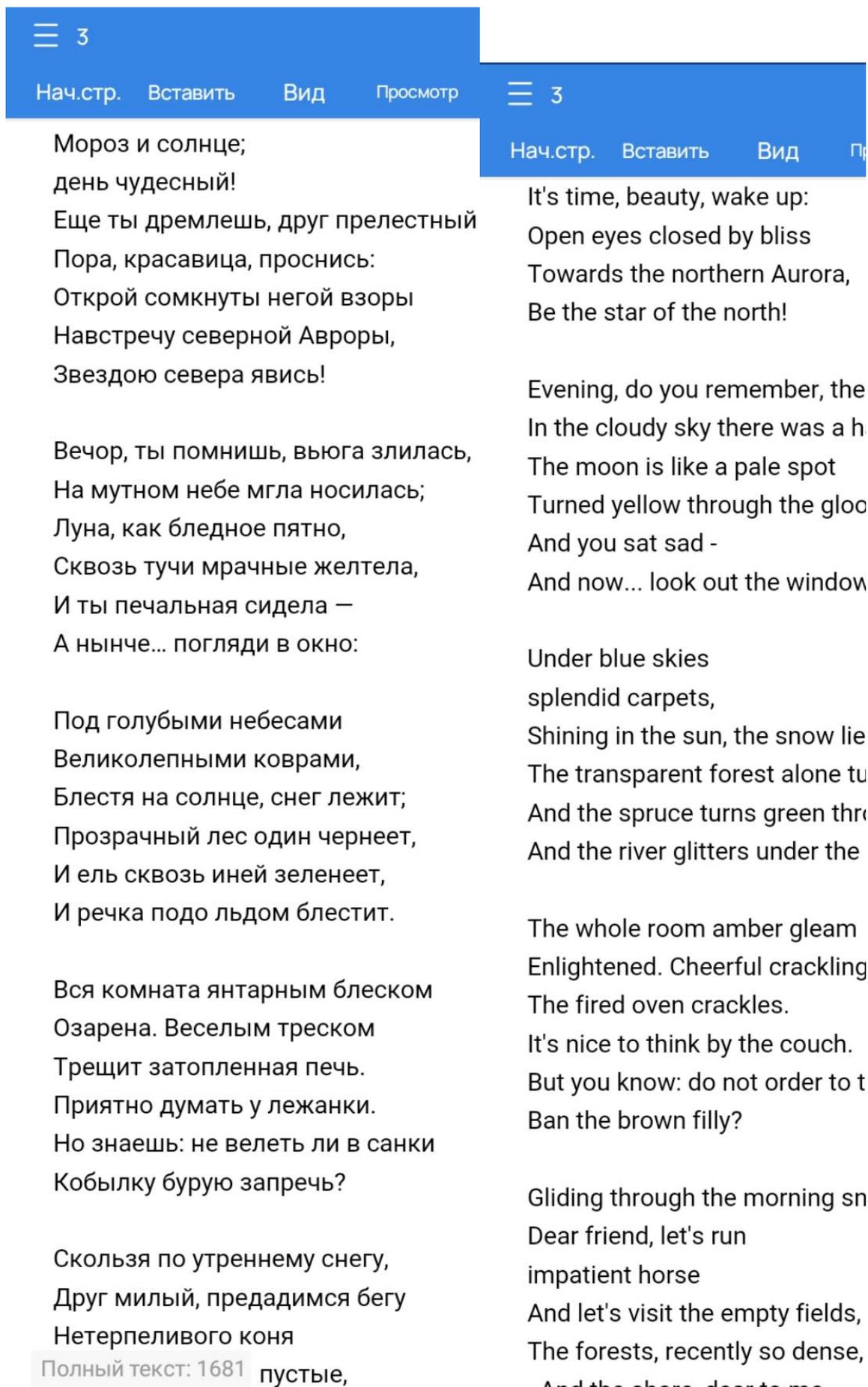


Рис. 26. Чтение текста на русском и английском языках из текстового файла 3.txt.

Если добавить несколько символов, например, три пробела перед символом-меткой в последней строке, как показано на Рис. 27.

proba15.cxx*	proba2.cxx	proba5.cxx
23	<pre> навестим поля пустые,\nЛеса, недавно столь густые,\nИ берег, милый для меня.\n\nПеревод\n\nFrost and sun;\nwonderful day!\nYou are still dozing, lovely friend -\nIt's time, beauty, wake up:\nOpen eyes closed by bliss\nTowards the northern Aurora,\nBe the star of the north!\n\nEvening, do you remember, the blizzard was angry, \nIn the cloudy sky there was a haze;\nThe moon is like a pale spot\nTurned yellow through the gloomy clouds,\nAnd you sat sad - \nAnd now... look out the window:\n\nUnder blue skies\nsplendid carpets,\nShining in the sun, the snow lies;\nThe transparent forest alone turns black,\nAnd the spruce turns green through the hoarfrost, \nAnd the river glitters under the ice.\n\nThe whole room amber gleam\nEnlightened. Cheerful crackling\nThe fired oven crackles.\nIt's nice to think by the couch.\nBut you know: do not order to the sled\nBan the brown filly?\n\nGliding through the morning snow\nDear friend, let's run\nimpatient horse\nAnd let's visit the empty fields,\nThe forests, recently so dense,\n And the shore, dear to me. 0\n"; </pre>	
24		
25	<pre> for(k=0;k<=n1;k++) </pre>	
26	<pre> { </pre>	
27	<pre> i=int(k/n); </pre>	
28	<pre> j=k%n; </pre>	
29	<pre> c1[i][j]=mes[k]; </pre>	
30	<pre> if(mes[k]==A8\ </pre>	

Рис. 27. В последнюю строку квазидиагональной текстовой матрицы добавлены 3 символа - пробела перед последним символом-меткой ноль.

В этом случае на консольной части номер столбца символа метки должен увеличиться на 3 позиции $j_0=16+3=19$. При этом значение многочлена Чебышева первого рода программа дает $\lambda = T_{19}(13)=173$. Как это видно из Рис. 28 на консольной части.

```

19
173

Мороз и солнце;
день чудесный!
Еще ты дремлешь, друг прелестный –
Пора, красавица, проснись:
Открой сомкнуты негой взоры
Навстречу северной Авроры,
Звездою севера явись!

Вечор, ты помнишь, вьюга злилась,
На мутном небе мгла носилась;
Луна, как бледное пятно,

```

Рис. 28. Перед последним символом-меткой ноль символьной строки добавлены 3 пробела.

```
Нетерпеливого коня
И навестим поля пустые,
Леса, недавно столь густые,
И берег, милый для меня.

Перевод

Frost and sun;
wonderful day!
You are still dozing, lovely friend -
It's time, beauty, wake up:
Open eyes closed by bliss
Towards the northern Aurora,
Be the star of the north!

Evening, do you remember, the blizzard was angry,
In the cloudy sky there was a haze;
The moon is like a pale spot
Turned yellow through the gloomy clouds,
And you sat sad -
And now... look out the window:

Under blue skies
splendid carpets,
Shining in the sun, the snow lies;
The transparent forest alone turns black,
And the spruce turns green through the hoarfrost,
And the river glitters under the ice.

The whole room amber gleam
Enlightened. Cheerful crackling
The fired oven crackles.
It's nice to think by the couch.
But you know: do not order to the sled
Ban the brown filly?

Gliding through the morning snow
Dear friend, let's run
impatient horse
And let's visit the empty fields,
The forests, recently so dense,
And the shore, dear to me.
[Program finished]
```

Рис. 29. Вывод дешифрованного текста на консоль с ключами $x=13, j_0=19$.
На Рис.29 программа правильно выводит ключ $j_0=19$. При этом, хотя дешифрование происходит также правильно, при этом шифр совершенно другой, вид текстового файла 2.txt полученный программой с ключами $x=13, j_0=19$ приведен на Рис. 30.

}蚩 r2 聿&鯨攸佻鬣 M 璫?沮鏡 Gz-----Разрыв страницы-----
 ◆ 蔞%嶋,逕遲 p9 藺耄 j+挖 /◆8 獯~架 RN 调 T 漢
 歎T 厶账 3€!~ ◆53 拈 6◆/頂*紕 辦 9 杓撈咄疇 6J◆盆[椎 P li◆-◆ 礅匣^,
 堆 滲 9 蝸鰈 z◆'◆ ◆;8 訐痲喏邸c◆;◆) gb 眈 x 昭 l+脞 遙:糝蝻 U
 柁 q1 脣 鍊?臻 F 調 4 鷓 U 悲 B◆/ 浞! |x 匣 c,逕◆/摧鑿@蠶,鯢 1 適&鋤攸
 YF 錫 吻悲 d 颺旨 K 卅蔥\$◆◆(嗒遯 9 莛%頤 G 择 O 浹 鈹@溧 D 越驕 T 姣
 ◆◆ uq◆ w 裊+脞-遲:瀾麵闖症
 麵 8 葑~<◆-髮 M 猝?嶼苟 k 觸忝 Pm 陳%◆甚 r2 蛭%6◆-◆G 钶 3K◆\$錫?
 讬&C◆:鷓 U 伯 C 燻得 跣藺%則 康◆/◆#鏗@钻 G 钶 9 磨 U 莖,鞞 F 仞 M
 棲 `[醢 Pk 觸緱 Z 理 裡,所 3 憤扃 J@贞 G 啖均箴,鞞 F 仞 M 喇 9 槌旱 G
 ◆◆-◆ 招%◆◆(◆!遯:潛 壑 H 啊;麋 V◆@潤.髡 M 攢?喉 e&l◆)f 肤 藪
 ◆5+綬獲◆9 論.轄 G 擷
 麋 U◆@伍 F 鈔 7 鸚 T◆ 盱◆1 猗◆ p 菽\$◆拈 f3 殞%錐愴嗜佟/磨 V 朝<
 匣圳 M 姬 9 楹鸪 L 猗肱 箇 ◆9+脞 /◆#鎮@蠻黥丰绋=^聞识 1 髡 N
 m◆◆-----Разрыв страницы-----
 j 敲^ 袂 ◆4+结催=9 訐%轄柁弄啜;楫◆!滯 oN▯B 晒傅 B si◆+ 城\$紐 ◆6 儼
 z9 蕞(壘榭 XN 喋淺瞳 H R◆4◆ Q 瓠燾◆-----Разрыв страницы-----
 j 敲 Z 理 裡,所 2 葱&鑿@株 G 邸 3 慶 U 绌 l 泝蹙 V 酹绳\傅 G si 戥 m 蔻
 \$縹 獯儼 w:▯\$師 G 囫 N 墉@擲悍 ◆◆;◆ fc◆-----Разрыв страницы-----
 i 拂 秭 梳,妳 /棧鎬 A 黏'鯢 N 塘 涯捨▯O XT 炁 ▯ 尅▯P zw 袂+搞儼 N9 藺耄
 ▯G 擷 麋◆a)苟 H nj 腓殞暉考 N |p◆)t 裡+笕溥 z9 論+◆▯G 帷;麋 V◆@H
 tk◆"◆}\傅 J {j 让] 盖\$◆拈 q2 織腌 O 愒 M 柁弄啜;廖韶 M 卜 Q◆> 湯浸▯
 厶账 3◆!~ 猓儼 n9 薄+◆▯HB
 K'椽槩 nj 髻^ 礫痲 b 偌 W ▯厥 b\$◆◆(◆◆?:栲筭 X 柁玎郇 D_\罍 y 質▯鍤
 b▯% 猶◆ w 裊{◆83 憤殞 x@鈴 4 鈞 N 绌 R◆
 ▯羗 M w 蚋◆<&ec 录 Lf 肯 睽 ◆9+玲儼{;杓%鞞 G 浴 2 麗 U 吠
 u◆ 激T~~◆%梯臚 厶褪 f\$祗 穀 2 翠'鑿@坳 G 藁<麋 V 灵\◆ cJ ▯◆)&◆▯
 骷 Uf 肢 豎 ◆={豨 3 拳賻 EA ,P 跖屁:\◆◆ 猿▯U 瑯
 禳, tq 鲜 e\$◆◆(◆◆?:楛蝻 UG 冽堺 p◆恠罍 ◆◆ 穢
 襴-握 V p 藺%悻+脞 遙:工 z'◆ 鈞 O▯:愁鞞 GJ 鬣筭 ◆ 脛脰~\$価 糲
 策 袖掌 p5◆ 邛曦◆6 灑 L 顛 K 瑤 U

Рис.30. Параметры текстового файла 2.txt с ключами шифра x=13, j0=19.
 Сравнивая Рис. 30 с Рис.25 видно их различие, например, по первому и пятому символам шифра-текста в первой строке, так как различны ключи.

Далее нами написана программа с комментариями, реализующая на андроидном варианте языка C++ (компиляторе Cxx) алгоритм(65)-(67). В качестве тестового примера использовано известное стихотворение А.С. Пушкина “Зимнее утро”.

```
#include<stdio.h>
#include<math.h>
int f(int j0,int z)
{
int x, i,p[j0+1];
x=13;
p[0]=1;p[1]=x;
for(i=2;i<=j0;i++)
{
p[i]=(2*x*p[i-1]-p[i-2])%z;
if(p[i]<0)
{
p[i]=p[i]+z;
}
}
return p[j0];
}
int const n=50,n1=2700,m1=int(n1/n),z=256,n2=7;
int main()
{
int i,j,k,i1,i2,i0,j0,k0,nn;
int c1[m1+1][n], c2[m1+1][n],c22[m1+1][n], s1,y,x,a,
c3[m1+1][n],s,mes1[n1+1],kod[m1+1][n];
char * mes="\nМороз и солнце; \nдень чудесный!\nЕще ты дремлешь, друг
прелестный —\nПора, красавица, проснись:\nОткрой сомкнуты негой взоры\nНавстречу
северной Авроры,\nЗвездою севера явись!\n\nВечор, ты помнишь, вьюга злилась,\nНа
мутном небе мгла носилась;\nЛуна, как бледное пятно,\nСквозь тучи мрачные желтела,\nИ
ты печальная сидела —\nА нынче... погляди в окно:\n\nПод голубыми
небесами\nВеликолепными коврами,\nБлестя на солнце, снег лежит;\nПрозрачный лес
один чернеет,\nИ ель сквозь иней зеленеет,\nИ речка подо льдом блестит.\n\nВся комната
янтарным блеском\nОзарена. Веселым треском\nТрещит затопленная печь.\nПриятно
думать у лежанки.\nНо знаешь: не велеть ли в санки\nКобылку бурую запречь?\n\nСкользя
по утреннему снегу,\nДруг милый, предадимся бегу\nНетерпеливого коня\nИ наведем
поля пустые,\nЛеса, недавно столь густые,\nИ берег, милый для меня.\n\nПеревод\n\nFrost
and sun;\nwonderful day!\nYou are still dozing, lovely friend -\nIt's time, beauty, wake
up:\nOpen eyes closed by bliss\nTowards the northern Aurora,\nBe the star of the
north!\n\nEvening, do you remember, the blizzard was angry,\nIn the cloudy sky there was a
haze;\nThe moon is like a pale spot\nTurned yellow through the gloomy clouds,\nAnd you sat
sad -\nAnd now... look out the window:\n\nUnder blue skies\nsplendid carpets,\nShining in the
sun, the snow lies;\nThe transparent forest alone turns black,\nAnd the spruce turns green
through the hoarfrost,\nAnd the river glitters under the ice.\n\nThe whole room amber
gleam\nEnlightened. Cheerful crackling\nThe fired oven crackles.\nIt's nice to think by the
couch.\nBut you know: do not order to the sled\nBan the brown filly?\n\nGliding through the
morning snow\nDear friend, let's run\nimpatient horse\nAnd let's visit the empty fields,\nThe
forests, recently so dense,\nAnd the shore, dear to me. 0\n";

for(k=0;k<=n1;k++)
{
i=int(k/n);
```

```

        j=k%n;
        c1[i][j]=mes[k];
        if(mes[k]==48)
        {
            j0=j;i0=i;
        }
    }
    a=f(j0,z);
    printf("%d \n",j0); // column j0
    printf("%d \n",a); //Chebishev mnogochlen
    // return 0;
    k0=(i0)*n+j0; // number sibols
    char shelf1[k0];
    for(i=0;i<=i0+1;i++)
    for(j=0;j<=n-1;j++)
    {
        k=i*n+j;
        if(k<=k0)
        {
            kod[i][j]=(i+j)%z; //code matrix
        }
    }
    for(i=0;i<=i0+1;i++)
    for(j=0;j<=n-1;j++)
    {
        printf("%c",c1[i][j]);
        if(c1[i][j]==48)
        {
            i=i0+1;j=n-1;
        }
    }
    for(i=0;i<=i0+1;i++)
    for(j=0;j<=n-1;j++)
    {
        k=i*n+j;
        if(k<=k0)
        {
            c2[i][j]=(a*kod[i][j]+c1[i][j])%z; //Encryption
        } }
    FILE*file;
    file=fopen("1.txt","w");
    for(i=0;i<=i0;i++)
    for(j=0;j<=n-1;j++)
    {
        k=i*n+j;
        if(k<=k0)
        {
            fprintf(file,"%c",c1[i][j]);
            printf("%c",c1[i][j]);
        }
        if(k==k0)
        {

```

```

                i=i0;j=n-1;
            }
            file=fopen("2.txt","w");
for(k=0;k<=k0;k++)
{
    i=int(k/n);
    j=k%n;
    fprintf(file,"%c", c2[i][j]); // Write to file 2.txt
}
    fclose(file);
    file=fopen("2.txt","rb");
for(k=0;k<=k0;k++)
{
    i=int(k/n);
    j=k%n;
nn=fread( shelf1 , sizeof(char), k0, file); // Reading from a file 2.txt
    c22[i][j]=shelf1[k];
}
for(k=0;k<=k0;k++)
{
    i=int(k/n);
    j=k%n;
    c2[i][j]=c22[i][j];
}
for(i=0;i<=i0+1;i++)
    for(j=0;j<=n-1;j++)
    {
        k=i*n+j;
        if(k<=k0)
        {
            c3[i][j]=(c2[i][j] -a*kod[i][j])%z; ; //Decryption
            if(c3[i][j]<0)
        {
            c3[i][j]=c3[i][j]+z;
        } } }
printf("\n");
file=fopen("3.txt","w");
for(k=0;k<=k0-1;k++)
{
    i=int(k/n);
    j=k%n;
    fprintf(file,"%c", c3[i][j]); // Write cipher to file 3.txt
    printf("%c", c3[i][j]);
}
}

```

Оказывается, что на множестве правых квазипрямоугольных матриц, у которых число строк не менее числа столбцов ($m \geq n$) можно ввести операцию умножения правых квазипрямоугольных правых матриц $C=A(m \times l) * B(l \times n)$ по формуле

$$\begin{cases} c_{i,j} = \left(\sum_{k=0}^l a_{i,k} b_{k,j} \right) (\text{mod } p), A(m \times l), B(l \times n) \\ a_{i,k} b_{k,j} = \begin{cases} a_{i,k} b_{k,j}, & \text{if } a_{i,k} b_{k,j} \text{ defined} \\ 0, & \text{if not } a_{i,k} b_{k,j} \text{ defined} \end{cases} \end{cases} \quad (68)$$

То есть, в формуле (68) слагаемое присутствует в сумме тогда и только тогда, если числа $a_{i,k}$, $b_{k,j}$ – элементы правых квазипрямоугольных матриц определены. Если в сумме(68) все слагаемые отсутствуют из-за того, что ни одно слагаемое не определено, то элемент $c_{i,j}$ записать невозможно и позиция в матрице C с индексами (i,j) пустая.

Более того, используя формулу(68) на классе квазиквадратных матриц можно находить с помощью присоединенной матрицы обратную матрицу к данной квазиквадратной. Квазиквадратной единичной матрицей назовем матрицу, у которой главная диагональ состоит из единиц, а остальные элементы нули. Рассмотрим пример вычисления обратной матрицы к правой квазиквадратной матрице A . В присоединенной матрице справа приписываем блок в виде правой квазиединичной матрицы той же размерности, что и исходная матрица A . Далее элементарными эквивалентными преобразованиями приводим присоединенную матрицу к виду, в котором слева находится квазиквадратная единичная матрица, а справа квазиквадратная матрица обратная к исходной.

$$A = \left(\begin{array}{cc|c} 1 & 2 & 0 \\ 3 & & 1 \end{array} \right); A = \left(\begin{array}{ccc} 1 & 2 & 0 \\ & 3 & 1 \end{array} \right) \xrightarrow{2/3} \left(\begin{array}{ccc} 1 & 2 & 0 \\ & 1 & 1/3 \end{array} \right) \xrightarrow{1-2 \cdot 2} \left(\begin{array}{ccc} 1 & 0 & 1 \\ & 1 & 1/3 \end{array} \right) \Leftrightarrow$$

$$A^{-1} = \left(\begin{array}{cc|c} 1 & -2/3 & 0 \\ & 1/3 & 1 \end{array} \right) \quad A \cdot A^{-1} \stackrel{(68)}{=} \left(\begin{array}{cc} 1 & 2 \\ & 3 \end{array} \right) \cdot \left(\begin{array}{cc} 1 & -2/3 \\ & 1/3 \end{array} \right) = \left(\begin{array}{cc} 1 \cdot 1 & 1 \cdot (-2/3) + 2 \cdot (1/3) \\ & 3 \cdot (1/3) \end{array} \right) = \left(\begin{array}{cc} 1 & 0 \\ & 1 \end{array} \right) = I$$

$$A^{-1} \cdot A \stackrel{(68)}{=} \left(\begin{array}{cc} 1 & -2/3 \\ & 1/3 \end{array} \right) \cdot \left(\begin{array}{cc} 1 & 2 \\ & 3 \end{array} \right) = \left(\begin{array}{cc} 1 \cdot 1 & 1 \cdot 2 + (-2/3) \cdot 3 \\ & (1/3) \cdot 3 \end{array} \right) = \left(\begin{array}{cc} 1 & 0 \\ & 1 \end{array} \right) = I$$

Пример 1 показывает, что квазиквадратные матрицы и обратные к ним коммутируют друг с другом и произведение матриц в любом порядке равно квазиквадратной единичной матрице. $A^{-1} \cdot A = A \cdot A^{-1} = I$.

Если в матричном уравнении присутствует квазиквадратная невырожденная матрица, то второй множитель по известной матрице в правой части можно найти по формуле

$$A \cdot B = C \Leftrightarrow A = C \cdot B^{-1} \tag{69}$$

Рассмотрим пример 2.

$$A = \left(\begin{array}{cc} 1 & 2 \\ 3 & 2 \\ & 1 \end{array} \right), B = \left(\begin{array}{cc} 1 & 2 \\ & 3 \end{array} \right), B^{-1} = \left(\begin{array}{cc} 1 & -2/3 \\ & 1/3 \end{array} \right), C = \left(\begin{array}{cc} 1 & 8 \\ 3 & 12 \\ & 3 \end{array} \right), A \cdot B = C \Leftrightarrow$$

$$A = C \cdot B^{-1} = \left(\begin{array}{cc} 1 & 8 \\ 3 & 12 \\ & 3 \end{array} \right) \cdot \left(\begin{array}{cc} 1 & -2/3 \\ & 1/3 \end{array} \right) = \left(\begin{array}{cc} 1 \cdot 1 & 1 \cdot (-2/3) + 8 \cdot (1/3) \\ 3 \cdot 1 & 3 \cdot (-2/3) + 12 \cdot (1/3) \\ & 3 \cdot (1/3) \end{array} \right) = \left(\begin{array}{cc} 1 & 2 \\ 3 & 2 \\ & 1 \end{array} \right)$$

Сравнивая исходную правую квазипрямоугольную матрицу $A(3 \times 2)$ с вычисленной по правилам матричного умножения с учетом формулы(68) видим, что матрицы совпадают.

Таким образом, на классе правых квазиквадратных матриц можно складывать матрицы, умножать их на действительное число с нулевым элементом пространства по групповой операции сложения – квазиквадратной нуль-матрицей той же размерности. Далее на классе правых невырожденных квазиквадратных матриц можно вычислять обратные матрицы(кроме обратной к нуль-матрице), можно умножать квазипрямоугольные матрицы с подчиненными размерностями. Если ввести по групповой операции умножения единичный элемент - квазиквадратную единичную матрицу, то множество правых квазиквадратных невырожденных матриц одной размерности с указанными нуль - элементом и единичным элементом можно наделить алгеброй со структурой кольца на множестве действительных чисел с группой по сложению и полугруппой по умножению.

В заключение оценим мощность пространства ключей кодовой матрицы и 2 дополнительных ключей для стихотворения А.С. Пушкина Зимнее утро в примере 8. Каждый символ ключей $1683=1681+2$ может принимать любой из 256 символов клавиатуры. Русский язык имеет 33 малых и 33 больших букв алфавита. Английский алфавит имеет 26 больших и 26 малых букв, то есть $33*2+26*2=118$ различных символов, что более чем 2 раза меньше чем число символов 256. Длина текста 1681 символов. Поэтому искомая мощность пространства ключей равна $N=118^{1683}=9,49*10^{3486}$. Если компьютер методом перебора проверяет 10000000000 ключей в секунду (эта скорость сопоставима с тактовой частотой современных ядер процессоров), то понадобится для перебора число лет $3*10^{3479}$. Если все компьютеры в мире начнут перебор одновременно (их число оценим как число человек на планете, умноженное на $100\sim 10^{13}$) . тогда все компьютеры мира переберут пространство ключей за число лет - $3*10^{3466}$.

Отметим также замечательные работы Уральской математической школы из Пермского государственного национального исследовательского университета, специалистов по алгебраической теории чисел, информационной безопасности и прикладной математике[5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24].

Приведенные 8 примеров шифрования целых чисел, целочисленных векторов и прямоугольных матриц (квазипрямоугольных матриц) показывают огромные возможности криптографии и богатство различных методов шифрования над полем Z_p . Результаты данной работы могут быть продолжены в работах[25],[27],[28],[29].

Литература

1. Кострикин, А. И. Линейная алгебра и геометрия: учебное пособие / А. И. Кострикин, Ю. И. Манин; А. И. Кострикин, Ю. И. Манин. – Изд. 4-е, стер.. – Санкт-Петербург [и др.] : Лань, 2008. – (Классическая учебная литература по математике). – ISBN 978-5-8114-0612-8. – EDN QJTGEF.
2. Лидовский В.В. Теория информации: Учебное пособие. – М.: Компания Спутник+, 2004. – 111 с. – ISSN 5-93406-661-7.
3. Пастухов, Д. Ф. Некоторые методы передачи QR-кода с помощью стеганографии / Д. Ф. Пастухов, Н. К. Волосова, А. К. Волосова // Мир транспорта. – 2019. – Т. 17, № 3(82). – С. 16-39. – DOI 10.30932/1992-3252-2019-17-3-16-39. – EDN RSGSLR.
4. Вакуленко, С. П. Способы передачи QR-кода в компьютерной стеганографии / С. П. Вакуленко, Н. К. Волосова, Д. Ф. Пастухов // Мир транспорта. – 2018. – Т. 16, № 5(78). – С. 14-25. – EDN YNQNT.
5. Черников, А. В. Одна из возможных реализаций модели менеджера паролей для ОС Андроид / А. В. Черников // Вестник Пермского университета. Математика. Механика. Информатика. – 2022. – № 1(56). – С. 38-47. – DOI 10.17072/1993-0550-2022-1-38-47. – EDN PNSEAY.
6. Чернов, П. К. Создание интегрированной модели данных из разнородных источников, содержащих цифровые следы / П. К. Чернов, Е. А. Рабчевский // Вестник Пермского университета. Математика. Механика. Информатика. – 2022. – № 2(57). – С. 81-87. – DOI 10.17072/1993-0550-2022-2-81-87. – EDN UYUSGT.
7. Разработка специальной классификации информационных активов в сфере информационной безопасности / А. В. Манжосов, И. П. Болодурина, В. С. Сабуров, Н. А. Долгушев // Вестник Пермского университета. Математика. Механика. Информатика. – 2022. – № 4(59). – С. 54-60. – DOI 10.17072/1993-0550-2022-4-54-60. – EDN ZHZWNB.

8. Нехорошева, Э. А. Построение модели протокола электронного голосования с возможностью проверки результата избирателями / Э. А. Нехорошева, А. П. Шкарапута // Вестник Пермского университета. Математика. Механика. Информатика. – 2022. – № 4(59). – С. 61-67. – DOI 10.17072/1993-0550-2022-4-61-67. – EDN QAMNYK.
9. Поторочина, К. Л. Безопасность применения IoT в сфере здравоохранения / К. Л. Поторочина, Е. Ю. Никитина // Вестник Пермского университета. Математика. Механика. Информатика. – 2022. – № 4(59). – С. 68-81. – DOI 10.17072/1993-0550-2022-4-68-81. – EDN FBHTIG.
10. Рабчевский, А. Н. Выявление признаков информационных операций на основе анализа начальной частоты публикации дубликатов / А. Н. Рабчевский, М. Ю. Карпов, Е. Г. Ашихмин // Вестник Пермского университета. Математика. Механика. Информатика. – 2022. – № 4(59). – С. 82-88. – DOI 10.17072/1993-0550-2022-4-82-88. – EDN TYPKBC.
11. Горячев, С. Н. Построение инструментальной модели для исследования системы "Компьютерный вирус-переносчик-автоматизированное рабочее место-локальная вычислительная сеть" / С. Н. Горячев, Н. С. Кобяков, С. Н. Костарев // Вестник Пермского университета. Математика. Механика. Информатика. – 2021. – № 1(52). – С. 53-56. – DOI 10.17072/1993-0550-2021-1-53-56. – EDN MСYXZQ.
12. Импортзамещение в сфере телекоммуникаций и IT-технологий / В. Л. Осипов, И. В. Жигалов, А. С. Лубянков [и др.] // Вестник Пермского университета. Математика. Механика. Информатика. – 2021. – № 1(52). – С. 70-74. – DOI 10.17072/1993-0550-2021-1-70-74. – EDN ZHKVOL.
13. Чернов, П. К. Модификация алгоритма на основе сети Фейстеля с добавлением элемента случайности в ключ шифрования / П. К. Чернов, А. П. Шкарапута // Вестник Пермского университета. Математика. Механика. Информатика. – 2021. – № 1(52). – С. 81-88. – DOI 10.17072/1993-0550-2021-1-81-88. – EDN MGBPSA.
14. Половицкий, Я. Д. Пермская алгебраическая школа С.Н. Черникова и ее предшественники / Я. Д. Половицкий // Вестник Пермского университета. Математика. Механика. Информатика. – 2021. – № 3(54). – С. 68-73. – DOI 10.17072/1993-0550-2021-3-68-73. – EDN PDKKGF.
15. Черников, А. В. Рекомендации по разработке менеджеров паролей для ОС Андроид / А. В. Черников // Вестник Пермского университета. Математика. Механика. Информатика. – 2021. – № 4(55). – С. 49-57. – DOI 10.17072/1993-0550-2021-4-49-57. – EDN HWPILN.
16. Ронзин, В. И. Разработка программного модуля поиска нарушений для интегрированной системы безопасности / В. И. Ронзин, Е. Ю. Никитина // Вестник Пермского университета. Математика. Механика. Информатика. – 2020. – № 1(48). – С. 69-73. – DOI 10.17072/1993-0550-2020-1-69-73. – EDN MSQOTG.
17. Тюрин, С. Ф. Три квантора / С. Ф. Тюрин // Вестник Пермского университета. Математика. Механика. Информатика. – 2020. – № 1(48). – С. 87-91. – DOI 10.17072/1993-0550-2020-1-87-91. – EDN MZNRKB.
18. Шимановская, М. В. Многопоточность на платформе.NET. Обзор средств / М. В. Шимановская, И. А. Муфтеев, Е. И. Илларионова // Вестник Пермского университета. Математика. Механика. Информатика. – 2020. – № 2(49). – С. 69-75. – DOI 10.17072/1993-0550-2020-2-69-75. – EDN CLCJAF.

19. Разработка элементов криптопроцессора с использованием отечественной САПР "Ковчег" / О. А. Зобнина, А. Н. Каменских, Г. К. Королев, С. Ф. Тюрин // Вестник Пермского университета. Математика. Механика. Информатика. – 2019. – № 2(45). – С. 60-66. – DOI 10.17072/1993-0550-2019-2-60-66. – EDN IYZAXK.
20. Боков, К. А. Компьютерное моделирование перколяции k-меров на квадратной решетке / К. А. Боков, М. М. Бузмакова // Вестник Пермского университета. Математика. Механика. Информатика. – 2018. – № 1(40). – С. 51-55. – DOI 10.17072/1993-0550-2018-1-51-55. – EDN ХМНХWP.
21. Александрова, Е. И. Модификация алгоритмов на основе сети Фейстеля посредством внесения избыточности с помощью кодов Хэмминга / Е. И. Александрова, А. П. Шкарапута // Вестник Пермского университета. Математика. Механика. Информатика. – 2018. – № 3(42). – С. 95-103. – DOI 10.17072/1993-0550-2018-3-95-103. – EDN VKVNHZ.
22. Разработка программного обеспечения для выявления источников общественноопасной информации в социальной сети "ВКонтакте" / А. А. Березин, Э. Ф. Гайнутдинов, А. С. Максимов, Е. Ю. Никитина // Вестник Пермского университета. Математика. Механика. Информатика. – 2018. – № 3(42). – С. 104-110. – DOI 10.17072/1993-0550-2018-3-104-110. – EDN VKVNIG.
23. Мустакимова, Я. Р. Создание базы данных и разработка архитектуры системы для выявления участников "групп смерти" / Я. Р. Мустакимова, Е. Ю. Никитина, А. Д. Турпанова // Вестник Пермского университета. Математика. Механика. Информатика. – 2018. – № 3(42). – С. 117-123. – DOI 10.17072/1993-0550-2018-3-117-123. – EDN YMJSYX.
24. Евстафьев, Е. О. Алгоритм динамической обфускации информации с ограничением количества попыток расшифровки, исполнения и просмотра на web-клиенте / Е. О. Евстафьев, С. Ф. Тюрин // Вестник Пермского университета. Математика. Механика. Информатика. – 2018. – № 4(43). – С. 56-59. – DOI 10.17072/1993-0550-2018-4-56-59. – EDN YRJEDJ.
25. Алгебраические методы шифрования : Учебное пособие к лекционным и практическим занятиям для студентов специальности 1-98 01 01 Компьютерная безопасность / Д. Ф. Пастухов, Н. К. Волосова, Ю. Ф. Пастухов [и др.]. – 3-е издание. – Москва, 2022. – 37 с. – EDN SNZVRK.
26. Бартенев О.В. Фортран для профессионалов. Математическая библиотека IMSL: Ч.3. – М.: ДИАЛОГ – МИФИ, 2001. – 368 с.
27. Моделирование систем. Лекции. Лабораторный практикум / Д. Ф. Пастухов, Ю. Ф. Пастухов, Н. К. Волосова [и др.]. – 3-е дополненное. – Новополюк, 2022. – 142 с. – EDN PYJINB.
28. Волосова Н.К., Пастухов Д.Ф., Волосов К.А. Методы расширения области применения методов математической физики//Международная конференция “Квазилинейные уравнения и обратные задачи”. QIPA conference handbook and proceedings. – М.: МФТИ, 2018. – С 20.
29. Пастухов Д.Ф., Пастухов Ю.Ф., Сеница П.Р. Шифрование данных на базе эллиптических кривых: Учебное пособие – Новополюк, ПГУ, 2016. – 72 с. (<http://elib.psu.by:8080/handle/123456789/16814>).

УДК 517. 66

Карлов Михаил Иванович(рецензент)
(Московский физико-технический университет)

Наталья Константиновна Волосова
(МГТУ им. Н.Э. Баумана)

Волосов Константин Александрович, Волосова Александра Константиновна
(Российский Университет Транспорта)

Дмитрий Феликсович Пастухов, Юрий Феликсович Пастухов
(Полоцкий университет)

Селиванов Виталий Александрович
(Полоцкий университет)

АЛГЕБРАИЧЕСКИЕ МЕТОДЫ ШИФРОВАНИЯ

2023