Министерство образования Республики Беларусь

Учреждение образования

«Полоцкий государственный университет»

Министерство науки и образования Российской Федерации

Московский государственный технический университет им. Н.Э. Баумана

Московский физико-технический университет

Российский университет транспорта

ПАСТУХОВ Д.Ф. (Полоцкий университет);
ВОЛОСОВА Н.К. (Московский государственный технический университет им. Н.Э. Баумана – Национальный исследовательский университет);
ПАСТУХОВ Ю.Ф. (Полоцкий университет)
ВОЛОСОВ К.А., ВОЛОСОВА А.К. (Российский университет транспорта)
КАРЛОВ М.И.(Московский физико-технический университет)

АЛГЕБРАИЧЕСКИЕ МЕТОДЫ ШИФРОВАНИЯ

Учебное пособие к лекционным и практическим занятиям
для студентов специальности

1-98 01 01 Компьютерная безопасность

Новополоцк, Москва

2022

УДК 519.66

Рецензенты:
А.А. Козлов, кандидат физико-математических наук, доцент, Заведующий кафедрой высшей математики и дифференциальных уравнений Полоцкого государственного университета

Пастухов Д.Ф., Волосова Н.К., Пастухов Ю.Ф.,Волосов К.А., Волосова А.К., Карлов М.И.

Алгебраические методы шифрования: Учебное пособие (третье издание). – Новополоцк, Москва. 2022.-37с.

В учебном пособии рассмотрено два примера преобразований $Z_p \to Z_p \times Z_p$ при шифровании и $Z_p \leftarrow Z_p \times Z_p$ при дешифровании. А также пример отображения $Z_p \times Z_p \times Z_p \to Z_p \times Z_p \times Z_p$ 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 $Z_p \to Z_p \times Z_p \times Z_p$, $Z_p \times Z_p \times Z_p \to Z_p$ Предложены методы шифрования ростками аналитических функций. Обобщен метод группового кодирования в поле остатков Z_2 на поле остатков группы Z_p с простым модулем р. Программы к примерам, написанным 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

Оглавление

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 $Z_p \to Z_p$, а при дешифровании обратное отображение $Z_p \leftarrow Z_p$. Здесь мы также рассмотрим два примера преобразований $Z_p \to Z_p \times Z_p$ при шифровании и обратно $Z_p \leftarrow Z_p \times Z_p$ при дешифровании. А также пример взаимно-однозначного аффинного отображения $Z_p \times Z_p \times Z_p \to Z_p \times Z_p \times Z_p$ 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 $Z_p \to Z_p \times Z_p \times Z_p$ при шифровании и обратного преобразования при дешифровании $Z_p \times Z_p \times Z_p \to Z_p$ ростками элементарных функций.

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

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

Авторы читали студентам курсы Математические основы криптографии, Теория информации, Теоретико-числовые аргументы информационной безопасности, используя также материал из данного пособия. Все программы к примерам в пособии написаны на языке FORTRAN, кроме последней (C++ является одним из распространенных языков в последнее время), поэтому программа на языке C++ может принести студентам наибольшую пользу.

Дизайн обложки учебного пособия объясняется тем, что его авторы благодарны указанным университетам г. Москвы, в которых они получили образование или работают.

Раздел 1

В разделе 1 использованы известные формулы разложения алгебраических полиномов на множители для шифрования текстовых данных над полем целочисленных остатков по простому модулю p.

Пример 1(Algebra 1).

Запишем известную алгебраическую формулу

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

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

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

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 $(R(x,y,n)(\mathrm{mod}\ p); Q(x,y,n)(\mathrm{mod}\ p))$

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

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

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

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

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

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

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

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

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

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

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

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

Дешифрование проводим по формуле(10), p простое.

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

Ниже написана программа с использованием алгоритма(7)-(10) на языке FORTRAN, в которой шифруется символьная фраза s="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,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
```

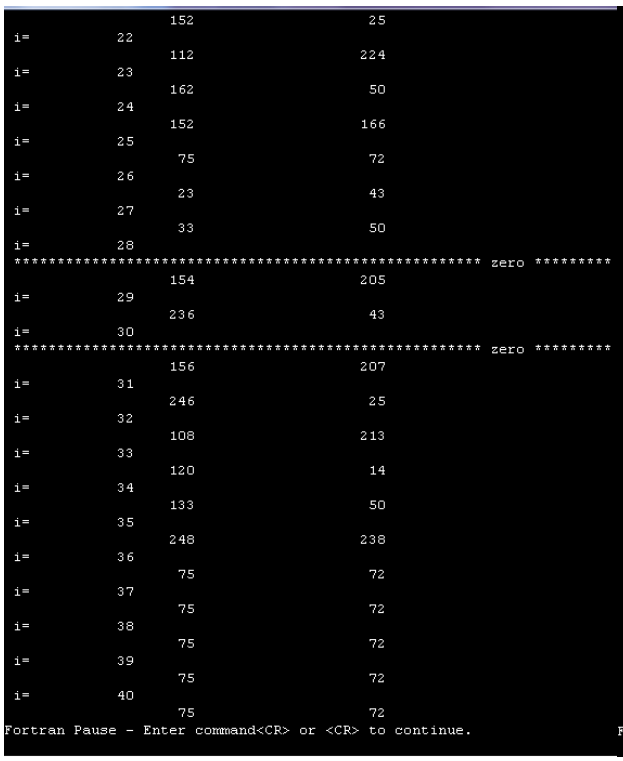


Рис.1

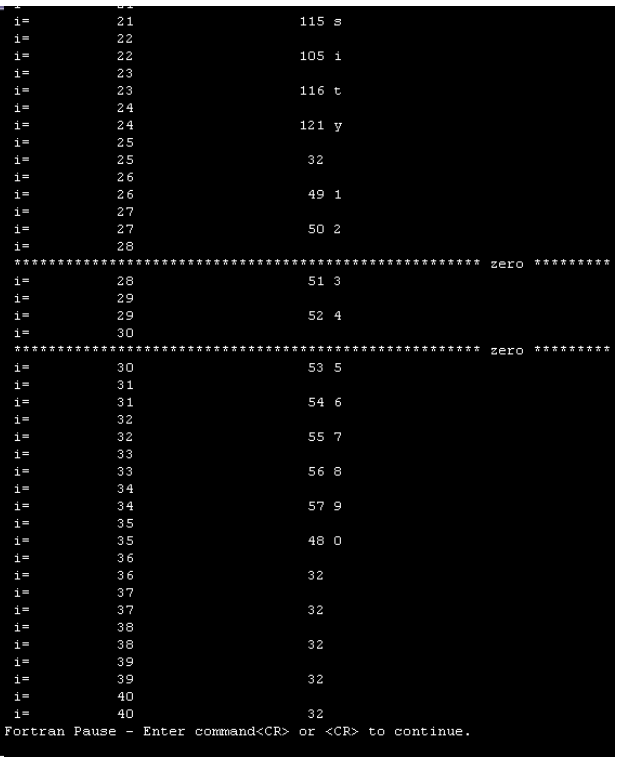


Рис.2

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 $T\approx 1.677\cdot 10^{11-3}c=1.677\cdot 10^8 c=1941 суток\approx 5{,}3 лет$. 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.

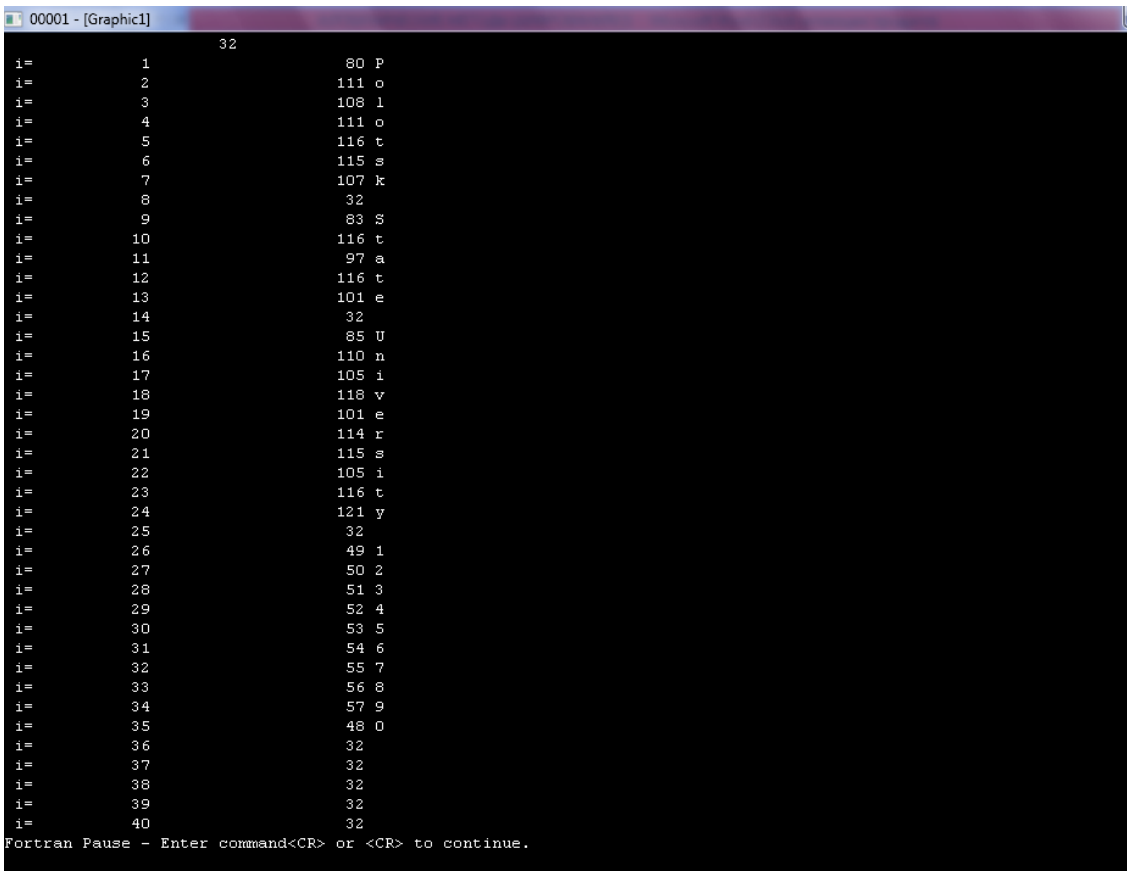


Рис.3. Пример дешифрования программой Algebra 1.

Пример 2(Algebra 2).

Запишем вторую алгебраическую формулу

$$y^{2k+1}+x^{2k+1}=(y+x)\left(y^{2k}-y^{2k-1}x+...+(-1)^{2k-1}yx^{n-2}+(-1)^{2k}x^{2k}\right)=(y+x)\sum_{i=0}^{2k}x^{i}y^{2k-i} \qquad (11)$$

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

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

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 $(R(x,y,n)(\mathrm{mod}\ p);Q(x,y,n)(\mathrm{mod}\ p))$

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

$$y\equiv x(\mathrm{mod}\,p)+R(x,y,n)(\mathrm{mod}\,p)\cdot Q^{-1}(x,y,n)(\mathrm{mod}\,p)\,|\,Q^{-1}(x,y,n)(\mathrm{mod}\,p)\cdot Q(x,y,n)\equiv 1(\mathrm{mod}\,p) \qquad (14)$$

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

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

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

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

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

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).

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

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

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

Дешифрование проводим по формуле(20), р простое.

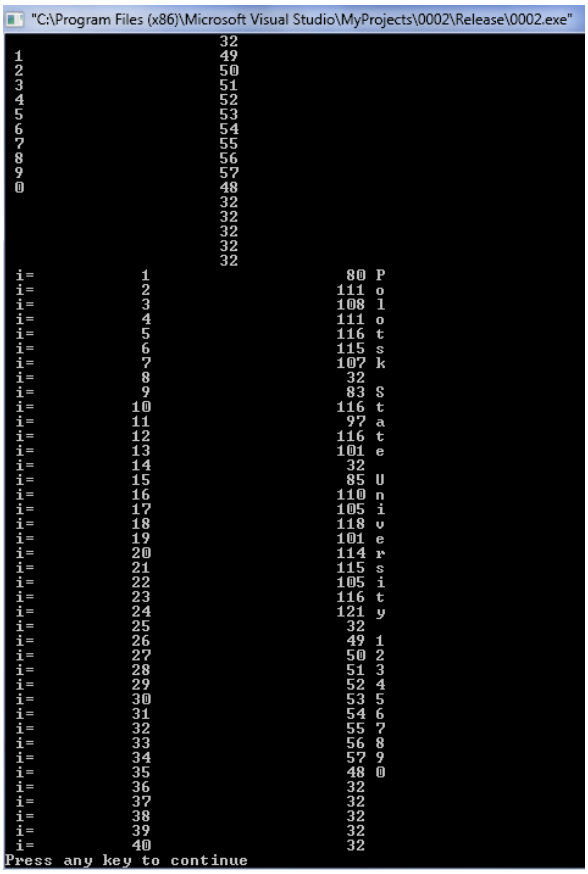
$$y = (R(x,y)+Q(x,y))(\bmod p)\cdot 2^{-1}(\bmod p) \quad (20)$$

Ниже написана программа с использованием алгоритма(17)-(20) на языке FORTRAN, в которой шифруется символьная фраза s="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,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=119;b=131
x=1;
do i=1,len
mas(i)=ichar(s(i:i))
print*,s(i:i),mas(i)
enddo
do i=1,len
y=mas(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)
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
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
```



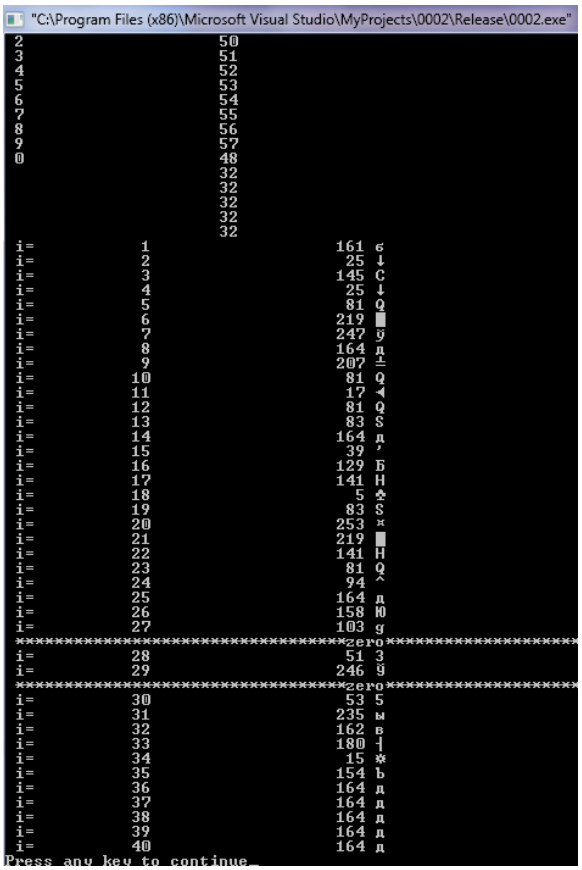


Рис.4. (n=10001) a=1119;b=131; x=103 Рис.5.(n=10000) a=1119;b=131; x=103

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 $\forall y \notin Y_0 : Q(x, y, n) \neq 0$. 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 $T \approx 1.677 \cdot 10^{11-3} c = 1.677 \cdot 10^8 c = 1941 суток \approx 5{,}3 лет$. 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).

Обозначим 12 ключей шифрования символами $A=\begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix}, b=\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$ (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 $Z_p \times Z_p \times Z_p \to Z_p \times Z_p \times Z_p$ 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

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

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

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

Найти обратное число к матрице системы(25)

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

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

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

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

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-a32*a23)-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
end if
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.

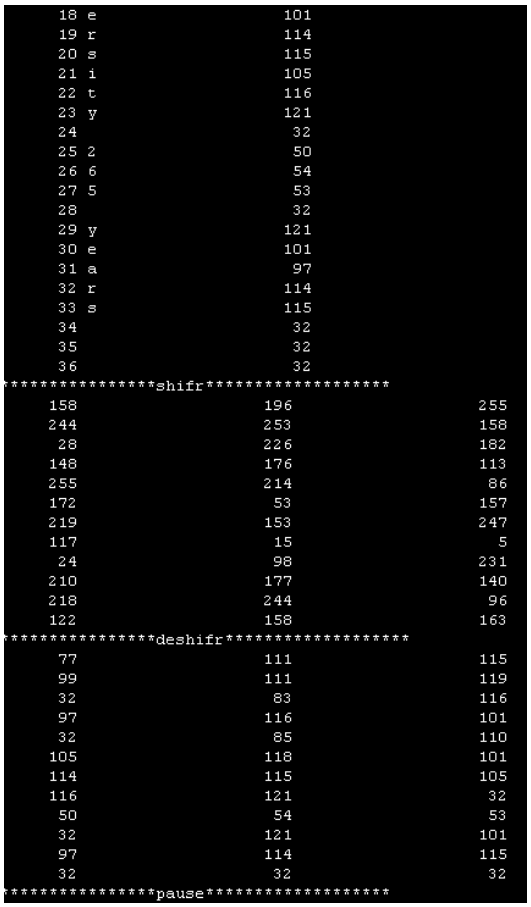


Рис.6

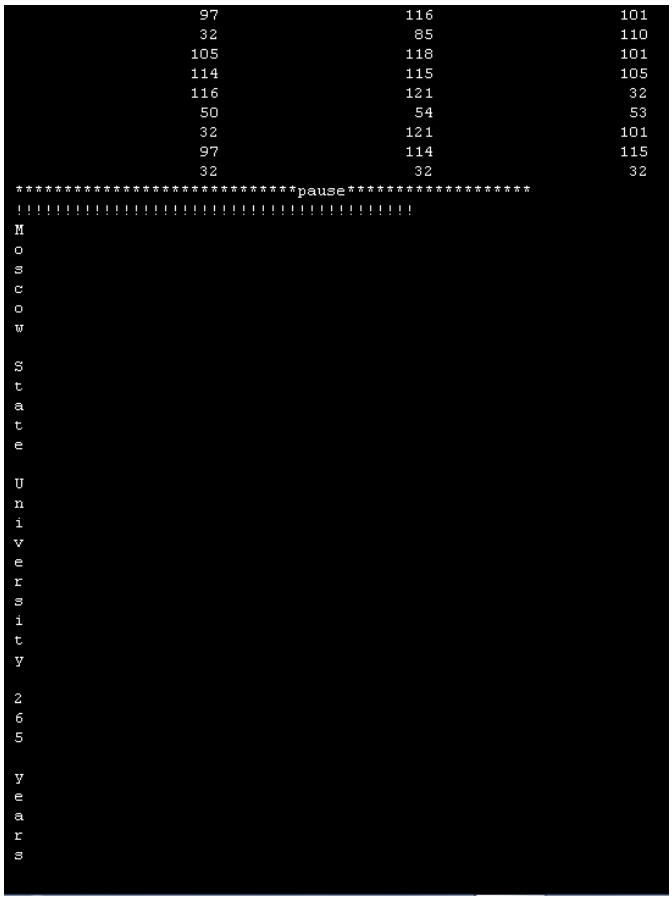


Рис.7

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 $R^n_{\exp(x)}(x)$

$$R^n_{\exp(x)}(x)=1+x+\frac{x^2}{2}+\frac{x^3}{6}+...+\frac{x^n}{n!},\exp(x)=1+x+\frac{x^2}{2}+\frac{x^3}{6}+...+\frac{x^n}{n!}+O\left(x^{n+1}\right) \quad (29)$$

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

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

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 $(R(x,n),Q(x,y,n),Z(x,y,n)(\mathrm{mod}\ p))$

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

То есть

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

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

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

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

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

Из формул(34),(35)следует, что для корректности необходимо условие $1\le x\le p-1$. 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*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
```

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

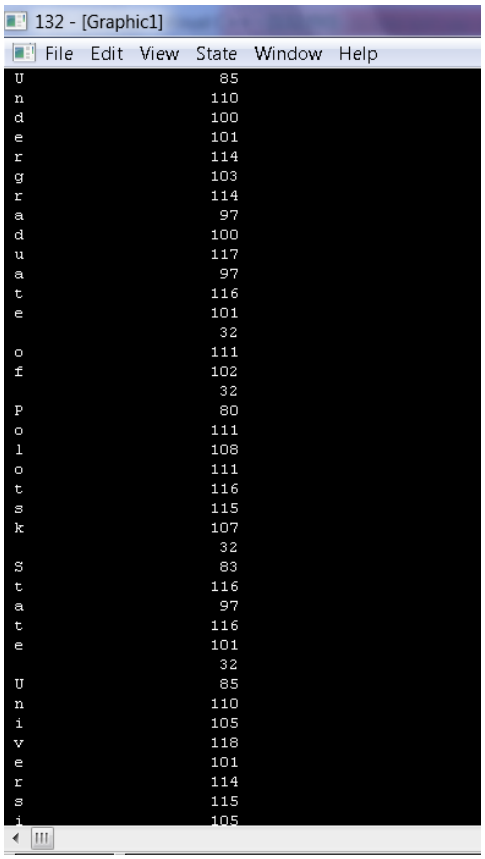


Рис.8

Исходная фраза для шифрования

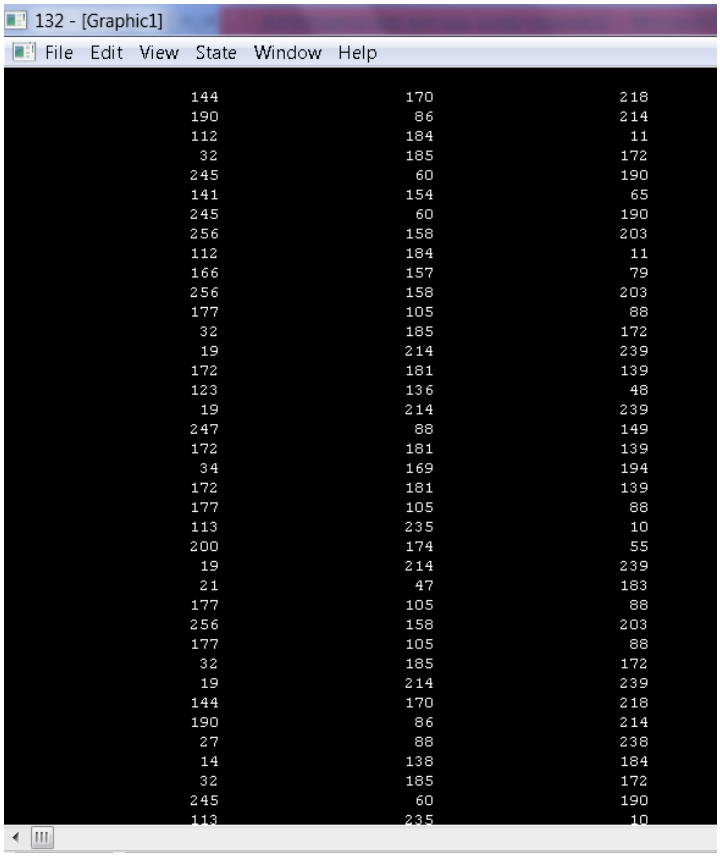


Рис.9

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

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 $Z_p \to Z_p \times Z_p \times Z_p$ шифровании и $Z_p \times Z_p \times Z_p \to Z_p$ при дешифровании.

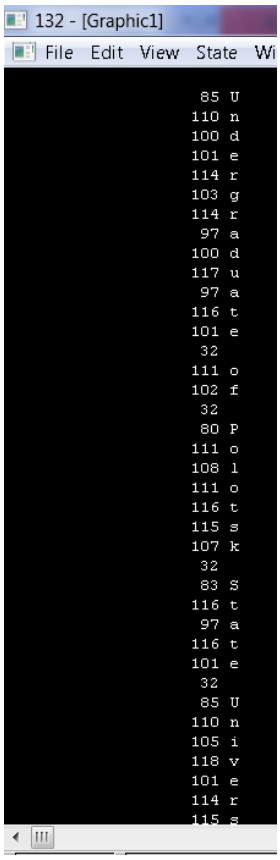


Рис.10

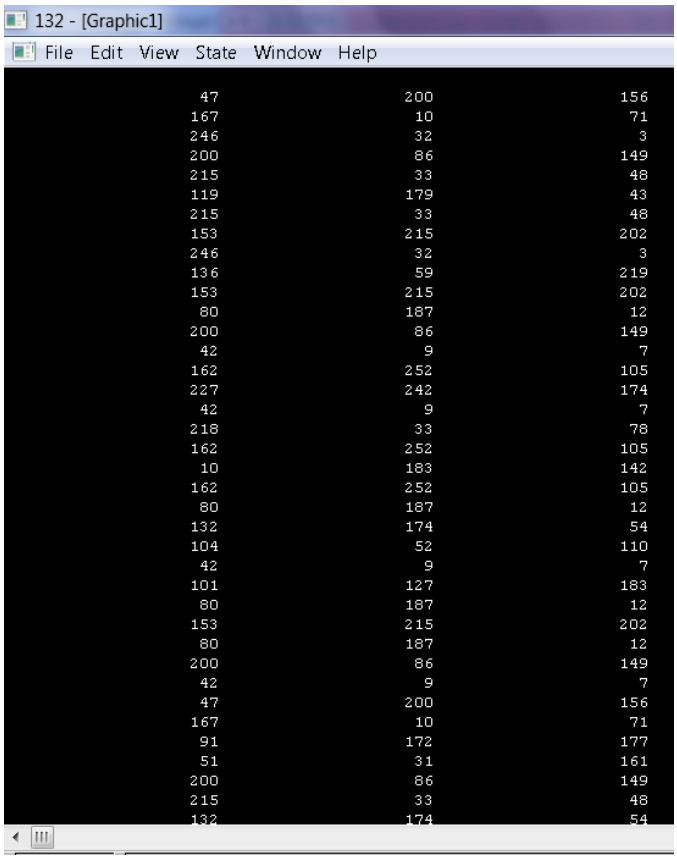


Рис.11

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

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

Сравнивая рисунки 9,11 видно, что росток одной и той же фразы при шифровании одной и той же функцией, нос 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

$$Z(x,n,a)=\frac{ax^{n-1}}{n!}(\mathrm{mod}\ p)=ax^{n-1}\cdot(n!)^{-1}(\mathrm{mod}\ p),(n!)^{-1}:(n!)^{-1}\cdot(n!)\equiv 1(\mathrm{mod}\ p),1\le a\le p-1 \qquad (36)$$

Тогда имеем формулы шифрования

$$(R(x,n),Q(x,n),Z(x,n)(\mathrm{mod}\ p))=\left(R^{n-1}_{y(x)}(x),R^{n}_{y(x)}(x),a\frac{x^{n-1}}{n!}\right)(\mathrm{mod}\ p) \qquad (37)$$

То есть

$$R(x,n)=R^{n-1}_{\exp(x)}(x)=\left(1+x+\frac{x^2}{2}+\frac{x^3}{6}+\ldots+\frac{x^{n-1}}{n-1!}\right)(\mathrm{mod}\ p)=\sum_{i=0}^{n-1}x^i\cdot(i!)^{-1}(\mathrm{mod}\ p),(i!)^{-1}:(i!)^{-1}(i!\cdot)\equiv 1(\mathrm{mod}\ p) \qquad (38)$$

$$Q(x,n)=R^{n-1}_{\exp(x)}(x)=\left(1+x+\frac{x^2}{2}+\frac{x^3}{6}+\ldots+\frac{x^{n}}{n!}\right)(\mathrm{mod}\ p)=\sum_{i=0}^{n}x^i\cdot(i!)^{-1}(\mathrm{mod}\ p),(i!)^{-1}:(i!)^{-1}(i!\cdot)\equiv 1(\mathrm{mod}\ p) \qquad (39)$$

Алгоритм дешифрования описывается формулой(40)

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

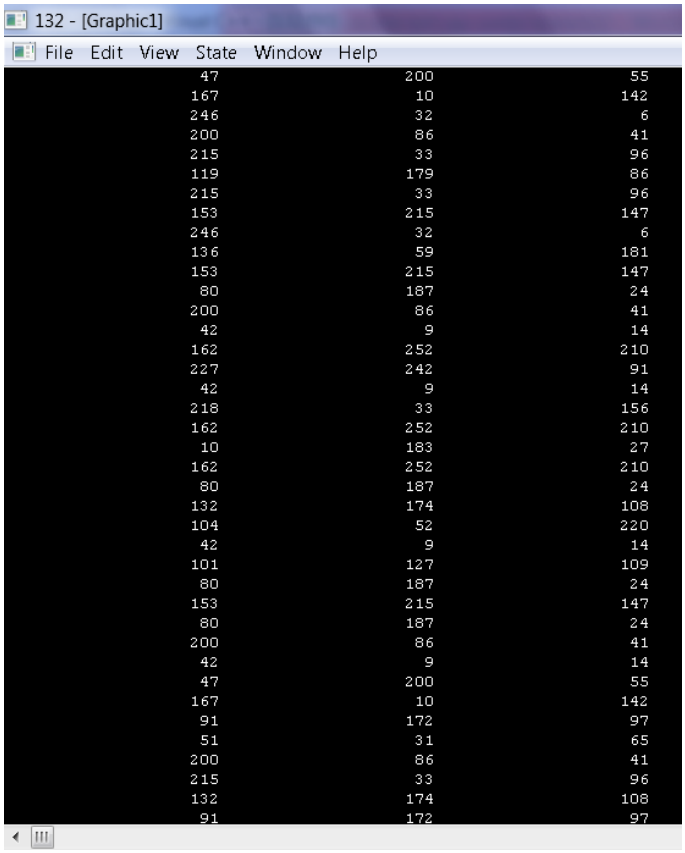


Рис.12

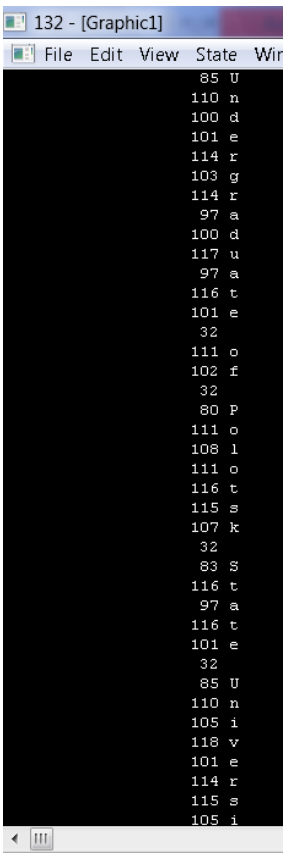


Рис.13

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 $R^{n}_{ch(x)}(x)$

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

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

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

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 $(R(x,2k),Q(x,2k),Z(x,2k)(\mathrm{mod}\ p))$

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

То есть

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

$$Q(x,2k)=R^{2k}_{ch(x)}(x)=\left(1+x^2+\frac{x^4}{4!}+...+\frac{x^{2k}}{(2k)!}\right)(\mathrm{mod}\ p)=\sum_{i=0}^{k}x^{2i}\cdot(2i!)^{-1}(\mathrm{mod}\ p),(2i!)^{-1}:(2i!)^{-1}(2i!\cdot)\equiv 1(\mathrm{mod}\ p) \quad (45)$$

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

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)

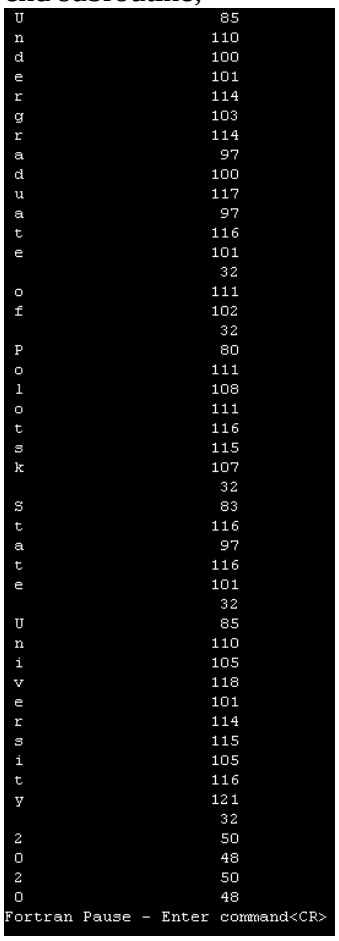
$$x\equiv(Q(x,2k)-R(x,2k))(Z(x,2k))^{-1}\cdot a(\mathrm{mod}\,p),(Z(x,2k))^{-1}:(Z(x,2k))^{-1}\cdot Z(x,2k)\equiv 1(\mathrm{mod}\,p) \quad (47)$$

Из формул(46)следует, что для корректности необходимо условие $1\le x\le p-1$. Далее следует программный код нашего алгоритма (41)-(47). Отметим, что мы сразу написали модифицированный 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;
```



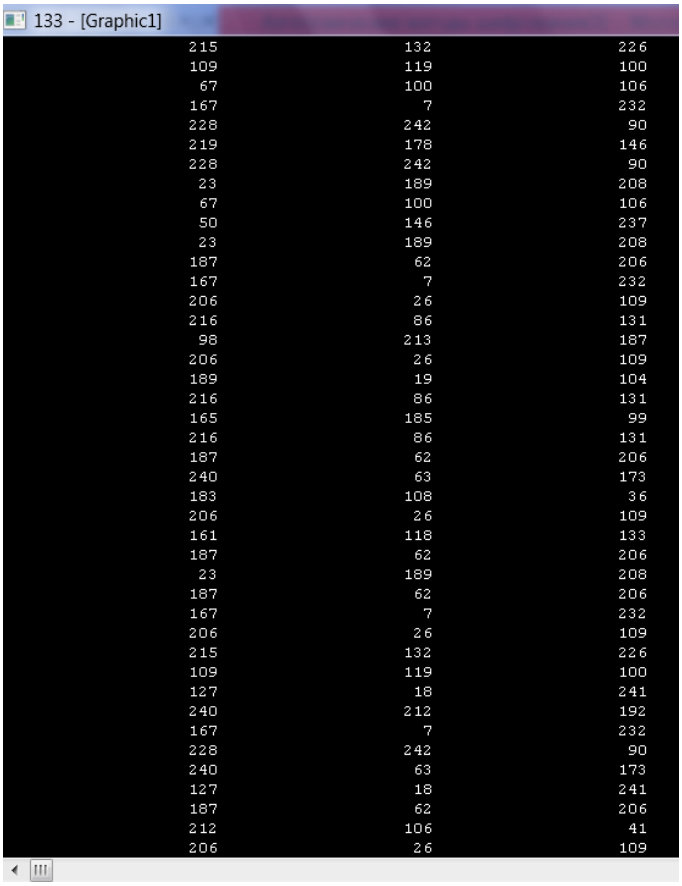


Рис.14 Рис.15

Результат шифрования с параметрами $(R(x,2k),Q(x,2k),Z(x,2k)(\mathrm{mod}\ p)), y(x)=ch(x), n=10, a=72$ $R_{ch(x)}^{n=10}, p=257$

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.

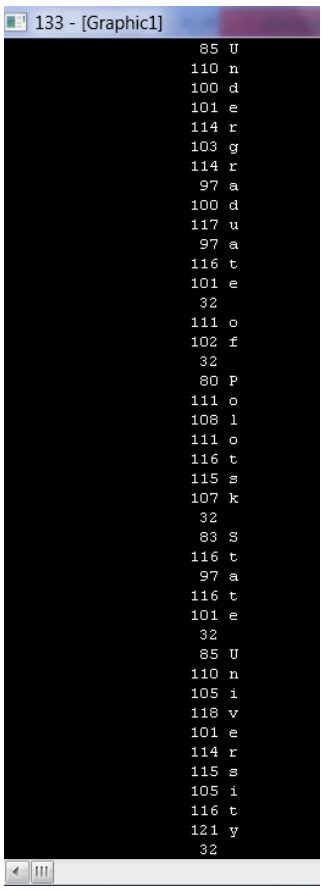


Рис.16 Результат дешифрования ростком гиперболического косинуса с параметрами $y(x)=ch(x), n=10, a=72$.

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 $R^{n}_{ch(x)}(x)$

$$R^{n=2k+1}_{sh(x)}(x)=\frac{1+x+\frac{x^2}{2}+\frac{x^3}{6}+...+\frac{x^{2k+1}}{2k+1!}-\left(1-x+\frac{x^2}{2}-\frac{x^3}{6}+...+(-1)^{2k+1}\frac{x^{2k+1}}{2k+1!}\right)}{2}=x+\frac{x^3}{3!}+\frac{x^5}{5!}+...+\frac{x^{2k+1}}{(2k+1)!} \quad (48)$$

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

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

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 $(R(x,2k+1), Q(x,2k+1), Z(x,y,2k)(\mathrm{mod}\ p))$

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

То есть

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

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

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

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)

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

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

```
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;
```

Раздел 3

Пример 7(Algebra 7).

В данном разделе мы опишем идею шифрования текста квадратными матрицами-ключами с элементами из поля $Z_p = \{0,1,2,..., p-1\}$. 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 $m = \left[\frac{k}{n}\right] + 1$ - 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.

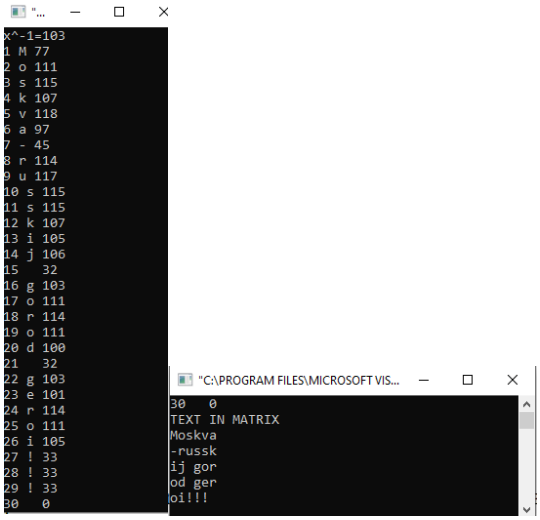


Рис.17. Исходная фраза текста"Moskva-russkij gorod geroi!!!" для шифрования записана в одномерном массиве.

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

Матричное шифрование проводим по формуле(56)

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

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

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

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

$$\overline{A}(m\times 2m)=\left(A(m\times m)\,|\,E(m\times m)\right)\sim\left(E(m\times m)\,|\,A^{-1}(m\times m)\right)(\text{mod}\,p) \tag{58}$$

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

$$2.\ k=\overline{1,m}: a_{k,j}^{new}\equiv a_{k,j}^{old}\cdot\left(a_{k,k}^{old}\right)^{-1}(\text{mod}\,p),\ j=\overline{1,2m},\left(a_{k,k}^{old}\right)^{-1}:\left(a_{k,k}^{old}\right)^{-1}\cdot a_{k,k}^{old}\equiv 1(\text{mod}\,p)$$

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

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

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

$$A\cdot U\equiv S(\text{mod}\,p)\Leftrightarrow U\equiv A^{-1}(\text{mod}\,p)\cdot S(\text{mod}\,p) \tag{61}$$

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

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

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

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.

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

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

На Рис.21 программой выполнена проверка $A^{-1}\cdot A\equiv 1(\mathrm{mod}\,p)$. 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 верны.

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 $\left(a_{k,k}^{old}\right)^{-1}(\mathrm{mod}\,p)$ к числу $a_{k,k}^{old}$ из множества остатков $\{1,2,...,\,p-1\}$ 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.

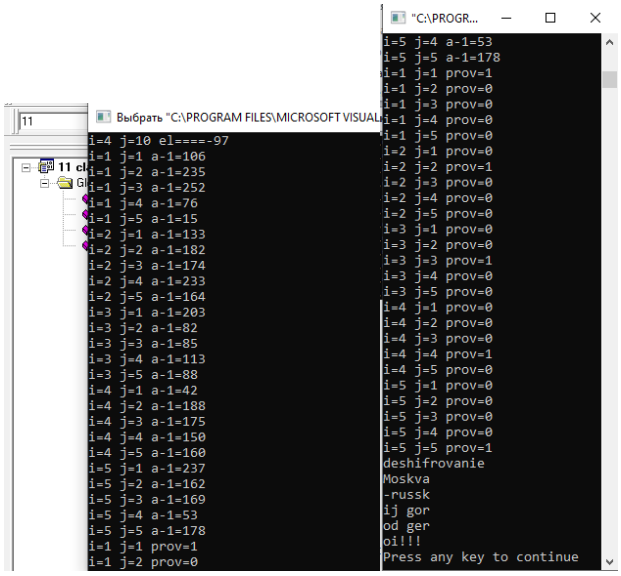


Рис.20. Для исходной матрицы(63) найдена обратная матрица $A^{-1}(\mathrm{mod}257)$.

Рис.21. Проверка условия $A^{-1}\cdot A\equiv 1(\mathrm{mod}\,p)$ 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

$65^{25}=2*10^{45}$. Если компьютер успевает перебирать матрицы-ключи под перехваченный шифр со скоростью 1 миллиард в секунду, то время перебора составит $7*10^{28}$лет. Даже перебор с помощью квантового компьютера такой мощности ключей практически невозможен.

Приведенные 7 примеров шифрования чисел, векторов и матриц над полем $Z_p = \{0,1,2,..., p-1\}$ показывает огромные возможности криптографии и богатство различных методов шифрования.

Литература

1. Кострикин А.И. Ведение в алгебру. Часть 1. Основы алгебры: Учебник для вузов. – 3-е изд.- М.:ФИЗМАТЛИТ,2004. – 272 С. – ISBN 5-9221-0487-X.
2. Лидовский В.В. Теория информации: Учебное пособие. – М.: Компания Спутник +,2004. – 111 с. –ISSN 5-93406-661-7.
3. Бартеньев О.В. Фортран для профессионалов. Математическая библиотека IMSL: Ч.1. – М.:ДИАЛОГ – МИФИ, 2001. – 448 с.
4. Бартеньев О.В. Фортран для профессионалов. Математическая библиотека IMSL: Ч.2. – М.:ДИАЛОГ – МИФИ, 2001. – 319 с.
5. Бартеньев О.В. Фортран для профессионалов. Математическая библиотека IMSL: Ч.3. – М.:ДИАЛОГ – МИФИ, 2001. – 368 с.
6. Пастухов Ю.Ф., Пастухов Д.Ф., Мередова М.Б. Динамическое кодирование: Учебное пособие. – Новополоцк: ПГУ,2019. – 19 с. (chttp:/ elib. psu.by:8080/ handle/123456789/23776).
7. Пастухов Ю.Ф. Вывод критерия Корсельта чисел Кармайкла из критерия связи чисел Кармайкла с функцией Кармайкла// Евразийское Научное Объединение. – 2021. № 12-1 (63). С. 31-34.
8. Пастухов Ю.Ф., Волосова Н.К., Волосов К.А., Волосова А.К., Пастухов Д.Ф., Пастухов А.Ю. Теорема о связи чисел Кармайкла с функцией Кармайкла// Евразийское Научное Объединение. –2021. № 6-1 (76). С. 50-53.
9. Вакуленко С.П., Волосова Н.К., Пастухов Д.Ф. Способы передачи QR-кода в стеганографии/ С.П. Вакуленко, Н.К. Волосова, Д.Ф. Пастухов //Мир транспорта. – 2018. Т.16. № 5(78). С. 14-25.
10. Пастухов Д.Ф., Волосова Н.К., Волосова А.К. Некоторые методы передачи QR-кода в стеганографии/ Д.Ф. Пастухов, Н.К. Волосова, А.К. Волосова //Мир транспорта. – 2019. Т.17. № 3(82). С. 16-39.
11. 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.
12. Пастухов Д.Ф., Пастухов Ю.Ф., Синица П.Р. Шифрование данных на базе эллиптических кривых: Учебное пособие – Новополоцк, ПГУ,2016. – 72 с. (http:/elib.psu.by:8080/handle/123456789/16814)
13. Волосова Н.К. Математическая модель динамики образования фибрина в аневризмах кровеносных капилляров// Евразийское Научное Объединение. –2021. № 12-1 (63). С. 11-17.
14. Волосова Н.К., Волосов К.А., Волосова А.К., Пастухов Д.Ф., Пастухов Ю.Ф., Сперанская О.А. Геометрический подход для качественного поиска конвективных

ячеек по температурному полю// Евразийское Научное Объединение. –2021. № 6-1 (76). С. 21-27.

15. Волосова Н.К., Волосов К.А., Волосова А.К., Карлов М.И., Пастухов Д.Ф., Пастухов Ю.Ф. Модифицированная формула Ньютона – касательных парабол на комплексной плоскости// Евразийское Научное Объединение. –2021. № 6-1 (76). С. 21-27.

УДК 517. 66

Дмитрий Феликсович Пастухов
(Полоцкий университет)

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

Юрий Феликсович Пастухов
(Полоцкий университет)

Волосов Константин Александрович, Волосова Александра Константиновна
(Российский университет транспорта)

Карлов Михаил Иванович
(Московский физико-технический университет)

АЛГЕБРАИЧЕСКИЕ МЕТОДЫ ШИФРОВАНИЯ

2022