

# Лабораторная работа № 1

## Основы теории чисел и их использование в криптографии

**Цель:** приобретение практических навыков выполнения операций с числами для решения задач в области криптографии и разработка приложений для автоматизации этих операций.

**Задачи:**

1. Закрепить теоретические знания по высшей арифметике.
2. Научиться практически решать задачи с использованием простых и взаимно простых чисел, вычислений по правилам модулярной арифметики и нахождению обратных чисел по модулю.
3. Ознакомиться с особенностями реализации готового программного средства L\_PROST и особенностями выполнения с его помощью операций над простыми числами.
4. Разработать приложение для реализации указанных преподавателем операций с числами.
5. Результаты выполнения лабораторной работы оформить в виде описания разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента.

### 1.1 ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

В основе современной криптографии лежит *теория чисел*.

Теория чисел или *высшая арифметика* – раздел математики, изучающий натуральные числа и иные похожие величины. В зависимости от используемых методов в теории чисел рассматривают несколько направлений. Нас будут интересовать вопросы *делимости целых чисел, вычисления наибольшего общего делителя (НОД), разложение числа на простые множители, малая теорема Ферма, теорема Эйлера, элементы теории вычетов*.

#### 1.1.1 Основные понятия и определения

Определение 1. Множество всех *целых чисел* (обозначим буквой  $\mathbb{Z}$ ) есть набор всех *действительных чисел* без дробной части:  $\{..., -3, -2, -1, 0, 1, 2, 3, ...\}$ .

Определение 2. *Натуральные числа* являются подмножеством целых чисел и образуют множество  $\mathbb{N}$ :  $\{1, 2, 3, ...\}$ .

Определение 3. *Делимость* – одно из основных понятий теории чисел.

Если для некоторого целого числа  $a$  и натурального числа  $b$  существует целое число  $q$ , такое, что  $bq=a$ , то говорят, что число  $a$  *делится* на  $b$ . В этом случае  $b$  называется *делителем* числа  $a$ , а  $a$  называется *кратным* числа  $b$ . При этом используются следующие обозначения:

$a:b$  –  $a$  делится на  $b$  или  $b|a$  –  $b$  делит  $a$ .

Из последнего определения следует, что:

- любое натуральное число является делителем нуля;
- единица является делителем любого целого числа;
- любое натуральное число является делителем самого себя.

Определение 4. Делитель  $a$  называется *собственным делителем* числа  $b$ , если  $1 < |a| < |b|$ , и *несобственным* – в противном случае.

Пример 1.  $4|20$ ; число 4 делит число 20, так как  $20 = 4 \cdot 5$ . При этом число 4 является собственным делителем числа 20.

Свойство 1 собственного делителя. Положительный наименьший собственный делитель составного числа  $n$  не превосходит  $\sqrt{n}$ .

Определение 5. Всякое целое число  $a$  можно представить с помощью положительного целого числа  $b$  равенством вида  $a = bq + r$ ,  $0 \leq r < b$ . Число  $q$  называется *неполным частным*, а число  $r$  – *остатком* от деления  $a$  на  $b$ .

### 1.1.2 Простые и составные числа

Каждое натуральное число, большее единицы, делится, по крайней мере, на два числа: на 1 и на само себя.

Если число не имеет делителей, кроме самого себя и единицы, то оно называется *простым*, а если у числа есть еще делители, то *составным*.

Определение 6. Натуральное число  $n$  называется *простым*, если  $n > 1$  и не имеет положительных делителей, отличных от 1 и  $n$ .

Простое число не делится без остатка ни на одно другое число.

Пример 2. Первые 10 простых чисел: 2, 3, 5, 7, 11, 13, 17, 19, 23 и 29. Простыми также являются числа 73, 2521, 2365347734339, 2756839 – 1. Количество простых чисел бесконечно велико.

Перечислим несколько ***важных свойств простых чисел***.

Свойство 1. Любое составное число представляется уникальным образом в виде произведения простых чисел; иначе еще говорят, что *разложение числа на простые множители однозначно*.

Это свойство вытекает из основной теоремы арифметики.

Основная теорема арифметики. Всякое натуральное число  $n$ , кроме 1, можно представить как произведение простых множителей:

$$n = p_1 \cdot p_2 \cdot p_3 \cdot \dots \cdot p_z, \quad z > 1. \quad (1.1)$$

Пример 3. Целое число  $1554985071 = 3 \cdot 3 \cdot 4463 \cdot 38713$  – произведение четырех простых чисел, два из которых совпадают.

Пример 4. Целое число  $39616304 = 2 \cdot 13 \cdot 7 \cdot 2 \cdot 23 \cdot 13 \cdot 2 \cdot 13 \cdot 2 \cdot 7 = 2 \cdot 2 \cdot 2 \cdot 2 \cdot 7 \cdot 7 \cdot 13 \cdot 13 \cdot 13 \cdot 23$ .

Порядок записи сомножителей после последнего знака равенства соответствует *канонической форме*.

Для того, чтобы представить относительно небольшое число в виде простых сомножителей, достаточно уметь делить числа столбиком. Однако при этом следует придерживаться некоторых простых правил. Для первого деления нужно выбрать наименьшее простое число большее 1, которое делит исходное число без остатка. Частное от первого деления также нужно разделить с учетом указанных ограничений. Процесс деления продолжаем до тех пор, пока частным не будет 1. Рассмотрим это на примерах.

Пример 5. Представить числа 144 и 39616304 в виде простых сомножителей. Порядок действий виден из нижеследующей иллюстрации.

144	2	1-й шаг	39616304	2
72	2	2-й шаг	19808152	2
36	2	.....	9904076	2
18	2		4952038	2
9	3		2476019	7
3	3		353717	13
1			27209	7
			3887	13
			299	13
			23	23
			1	

Таким образом,  $144 = 2 \cdot 2 \cdot 2 \cdot 2 \cdot 3 \cdot 3$ . Результат операций над другим числом был представлен выше в примере 4.

*Свойство 2.* Простых чисел бесконечно много, причем существует примерно  $n/\ln(n)$  простых чисел, меньших числа  $n$ .

*Свойство 3.* Наименьший простой делитель составного числа  $n$  не превышает  $\sqrt{n}$ , поэтому для проверки простоты числа достаточно проверить его делимость на 2 и на все нечетные (а еще лучше простые) числа, не превосходящие  $\sqrt{n}$ ; как видим, данное свойство коррелирует со свойством 1 собственного делителя.

Из соотношения  $n=qr$  натуральных чисел, больших единицы, следует, что либо  $p$ , либо  $q$  принадлежит отрезку от 2 до  $\sqrt{n}$ .

Поиск сомножителей числа  $n$  может вестись, например, перебором всех простых чисел до  $\sqrt{n}$ . Однако, если множители – большие простые числа, то на их поиск может потребоваться много времени.

Сложность решения задачи разложения больших чисел на простые сомножители, известной как проблема факторизации, определяет криптостойкость некоторых алгоритмов асимметричной криптографии, в частности алгоритма RSA.

*Свойство 4.* Любое четное число, большее 2, представимо в виде суммы двух простых чисел, а любое нечетное, большее чем 5, представимо в виде суммы трех простых чисел.

*Свойство 5.* Для любого натурального  $n$ , большего 1, существует хотя бы одно простое число на интервале от  $n$  до  $2n$ .

Определение 7. Натуральное число  $n$  называется составным, если  $n > 1$  и имеет, по крайней мере, один положительный делитель, отличный от 1 и  $n$ .

Единица не считается ни простым числом, ни составным.

*Пример 6.* Числа 17, 31 – простые, а числа 14, 15 — составные (14 делится на 2 и на 7, 15 делится на 3 и на 5).

Вернемся к собственному делителю.

*Свойство 2* собственного делителя. Положительный наименьший собственный делитель составного числа  $n$  есть простое число.

Так как простое число не делится ни на какое другое, кроме себя самого, очевидный способ проверки числа  $n$  на простоту – разделить  $n$  на все числа  $n-1$  и проанализировать наличие остатка от деления. Этот способ «в лоб» часто реализуется в компьютерных программах. Однако перебор может оказаться достаточно трудоемким, если на простоту нужно проверить число с количеством цифр в несколько десятков.

Существует правила, способные заметно сократить время вычислений [3].

*Правило 1.* Воспользоваться свойством 3 простых чисел (см. выше).

*Пример 7.* Проверим, является ли число 287 простым числом? Для этого найдем наименьший собственный делитель этого числа:

проверяем все простые числа от 2 до  $\sqrt{287}$ , т. е. от 2 до 13 (берется наибольшее простое число, не превышающее значение корня квадратного;  $\sqrt{287} \approx 16,94$ ).

получаем: ни одно из вышеуказанных простых чисел не является делителем числа 287. Таким образом, число 287 является простым.

*Правило 2.* Если последняя цифра анализируемого числа является четной, то это число заведомо составное.

*Правило 3.* Числа, делящиеся на 5, всегда оканчиваются пятеркой или нулем. Если младшим разрядом анализируемого числа являются 5 или 0, то такое число не является простым.

*Правило 4.* Если анализируемое число делится на 3, то и сумма его цифр тоже обязательно делится на 3.

*Пример 8.* Анализируемое число: 136827658235479371. Сумма цифр этого

числа равна:  $1 + 3 + 6 + 8 + 2 + 7 + 6 + 5 + 8 + 2 + 3 + 5 + 4 + 7 + 9 + 3 + 7 + 1 = 87$ . Это число делится на 3 без остатка:  $87 = 29 \cdot 3$ . Следовательно, и наше число тоже делится на 3 и является составным.

Правило 5. Основано на свойстве делимости на 11. Нужно из суммы всех нечетных цифр числа вычесть сумму всех четных его цифр. Четность и нечетность определяется счетом от младшего разряда. Если получившаяся разность делится на 11, то и анализируемое число тоже на него делится.

Пример 9. Анализируемое число: 2576562845756365782383. Сумма его четных цифр равна:  $8 + 2 + 7 + 6 + 6 + 7 + 4 + 2 + 5 + 7 + 2 = 56$ . Сумма нечетных:  $3 + 3 + 8 + 5 + 3 + 5 + 5 + 8 + 6 + 6 + 5 = 57$ . Разность между ними равна 1. Это число не делится на 11, а следовательно, 11 не является делителем анализируемого числа.

Правило 6. Основано на свойстве делимости на 7 и 13. Нужно разбить анализируемое число на тройки цифр, начиная с младших разрядов. Просуммировать числа, стоящие на нечетных позициях, и вычесть из них сумму чисел на четных. Проверить делимость результата на числа 7 и 13.

Пример 10. Анализируемое число: 2576562845756365782383. Перепишем его так: 2 576 562 845 756 365 782 383. Просуммируем числа, стоящие на нечетных позициях, и вычтем из них сумму чисел на четных:  $(383 + 365 + 845 + 576) - (782 + 756 + 562 + 2) = 67$ . Это число не делится ни на 7, ни на 13, а значит и делителями заданного числа они не являются.

Определение 8. Если два простых числа отличаются на 2, то их называют числами-близнецами.

Таких чисел не очень много. Например, ими являются 5 и 7, 29 и 31, 149 и 151.

Всякое натуральное число  $n > 1$  либо является простым числом, либо имеет простой делитель.

Воспользуемся перечисленными свойствами для определения простоты числа 2009. Это число не делится на 2 (так как оно нечетно), не делится также на 3 (сумма его цифр  $2+9=11$  не делится на 3), не делится и на 5. Воспользуемся далее свойством 6: попробуем разделить 2009 на 7; в результате получается целый результат: 287. Таким образом, получен ответ: число 2009 – составное.

Понятно, что в криптографии используются числа, проверка на простоту которых производится гораздо дольше, и для работы с этими числами требуются специальные программные средства. К вопросу проверки чисел на простоту мы еще вернемся. Здесь же отметим, что первый алгоритм нахождения простых чисел, не превышающих  $n$ , был придуман Эратосфеном во 2 в. до н. э. и известен сейчас как «решето Эратосфена». Его суть в последова-

тельном исключении из списка целых чисел от 1 до  $n$  чисел (или из сокращенного диапазона, например, от  $m$  до  $n$ ,  $1 < m \leq n$ ), кратных 2, 3, 5 и другим простым числам, уже найденным «решетом». Как видим, описанное выше свойство 2 простых чисел и положено в основу рассматриваемого алгоритма.

Для нахождения всех простых чисел не больше заданного числа  $n$  в соответствии с «*решетом Эратосфена*» нужно выполнить следующие шаги:

1. Выписать подряд все целые числа от двух (либо от  $m$ ) до  $n$  (2, 3, 4, ...,  $n$ ).

Пусть некоторая переменная (положим  $s$ ) изначально равна 2 – первому простому числу.

2. Удалить из списка числа от  $2s$  до  $n$ , считая шагами по  $s$  (это будут числа кратные  $s$ :  $2s, 3s, 4s, \dots$ ).

3. Найти первое из оставшихся чисел в списке, большее чем  $s$ , и присвоить значению переменной  $s$  это число.

4. Повторять шаги 2 и 3, пока возможно.

Пример 11. Примем  $n = 15$ .

Шаг 1. Выпишем числа от 2 до 15:

2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15.

Шаг 2. Удалим из списка числа с учетом  $s=2$ :

2, 3, 5, 7, 9, 11, 13, 15. В этом списке первое число, большее, чем  $s=2$ , это 3.

Текущему  $s$  присваивается новое значение:  $s = 3$ .

Шаг 3. Удалим из списка числа с учетом  $s=3$ :

2, 3, 5, 7, 11, 13, 15. В этом списке первое число, большее, чем  $s=3$ , это 5.

Текущему  $s$  присваивается новое значение:  $s = 5$ .

Шаг 4. Удалим из списка числа с учетом  $s=5$ :

2, 3, 5, 7, 13. В этом списке первое число, большее, чем  $s=5$ , это 7. Однако, в этом списке уже нет чисел, кратных текущему значению  $s$ , т.е. 7.

Таким образом, числа 2, 3, 5, 7, 13 являются простыми в диапазоне от 1 до 15. Как видим, количество таких чисел – 5.

Вспомним Свойство 2 *простых чисел* и посмотрим, как оно «работает» для нашего примера. Вычислим  $n/\ln(n) = 15/\ln 15 \approx 5,5$ . Результат (с учетом округления до целого) близок к истинному: количеству простых чисел от 1 до  $n = 15$ .

Пример 12. Найти все простые числа из промежутка [800; 830].

Воспользуемся Свойством 3 простых чисел и вычислим  $\sqrt{830} \approx 28,8$ , т. е. меньше 29. Запишем числа из заданного диапазона и удалим последовательно все числа, делящиеся на простые числа от 2 до 28. Такими простыми числами являются: 2, 3, 5, 7, 11, 13, 17, 19, 23. После выполнения всех операций в «решете» останутся числа: 809, 811, 821, 823.

Помимо рассмотренного алгоритма Эратосфена, который, понятно, является наименее эффективным, в настоящее время разработаны и используются другие алгоритмы. Описание и программную реализацию этих алгоритмов, можно найти, например, в известной и популярной у разработчиков криптографических приложений книге [4].

### 1.1.3 Взаимно простые числа и $\phi$ -функция

Понятие делимости чисел (см. Определение 3) является одним из важных в теории чисел. С этим понятием, а также с его производным – *общим делителем* (Определение 4) связаны другие важнейшие (в частности, для криптографии) понятия: *наибольшего общего делителя* (НОД) и *взаимно простых чисел*.

**Определение 9.** Наибольшее целое число, которое делит без остатка числа  $a$  и  $b$  называется *наибольшим общим делителем* этих чисел,  $\text{НОД}(a, b)$ .

**Пример 13.** Делителями числа  $a=24$  являются: 1, 2, 4, 6, 8, 12, 24; делителями числа  $b=32$  являются: 1, 2, 4, 8, 16, 32. Как видим,  $\text{НОД}(24, 32) = 8$ .

Понятно, что значение НОД можно вычислять для неограниченного ряда чисел.

Простым и эффективным средством вычисления  $\text{НОД}(a, b)$  является метод или *алгоритм Евклида* (примеры его использования приведены в [2]). В основе алгоритма лежит Определение 5. В соответствии с этим определением используется цепочка вычислений двумя исходными (начальными) числами:  $a$  и  $b$ :

$$a_i = b_i q_i + r_i, \quad 0 \leq r_i \leq b_i. \quad (1.2)$$

При  $i = 0$  в (1.2)  $a_i$  и  $b_i$  соответствуют как раз числам  $a$  и  $b$ . Последний ненулевой остаток ( $r_i, i \geq 0$ ) соответствует  $\text{НОД}(a, b)$ .

**Пример 14.** Пусть  $a = 1234, b = 54$ . Найти НОД.

$$\begin{aligned} 1234 &= 54 \cdot 22 + 46; \\ 54 &= 46 \cdot 1 + 8; \\ 46 &= 8 \cdot 5 + 6; \\ 8 &= 6 \cdot 1 + 2; \\ 6 &= 2 \cdot 3 + 0. \end{aligned}$$

Последний ненулевой остаток равен 2, поэтому  $\text{НОД}(1234, 54) = 2$ .

Чтобы найти НОД нескольких чисел (например,  $a, b, c$ ), достаточно найти НОД двух чисел (например,  $\text{НОД}(a, b) = d$ ) потом НОД полученного ( $\text{НОД}(a, b)$ ) и следующего числа ( $\text{НОД}(c, d)$ ) и т. д.

Таким образом, чтобы вычислить НОД  $k$  чисел, нужно последовательно вычислить  $(k-1)$  НОД. Последнее вычисление дает искомым результат.

**Определение 10.** Взаимно простыми являются целые числа, наибольший общий делитель которых равен 1.

**Пример 15.** Взаимно простыми являются числа 11 и 7, 11 и 4, хотя число 4 само по себе не является простым.

**Теорема 1.** Целые числа  $a$  и  $b$  взаимно просты тогда и только тогда, когда существуют такие целые  $u$  и  $v$ , что выполняется равенство

$$au + bv = 1. \quad (1.3)$$

**Теорема 2.** Если  $\text{НОД}(a, b) = d$ , то справедливо следующее соотношение (соотношение Безу):

$$au + bv = d. \quad (1.4)$$

Формула (1.4) называется также реализацией «расширенного алгоритма Евклида». Этот алгоритм состоит из двух этапов: собственно алгоритма Евклида и вычислений на основе обратных подстановок или последовательного выражения остатков в каждом из шагов предыдущего этапа с соответствующим приведением подобных на каждом шаге.

**Пример 16.** Для демонстрации обратимся к примеру 14, который составляет первый из указанных этапов. Ниже приведена таблица, из которой можно легко понять, как по алгоритму Евклида вычисляются остатки:

$$\begin{array}{ll} 1234 = 54 \cdot 22 + 46 & 46 = 1234 - 54 \cdot 22 \\ 54 = 46 \cdot 1 + 8 & 8 = 54 - 46 \cdot 1 \\ 46 = 8 \cdot 5 + 6 & 6 = 46 - 8 \cdot 5 \\ 8 = 6 \cdot 1 + 2 & 2 = 8 - 6 \cdot 1 \end{array}$$

Обратные подстановки или проход вверх начинаются от записи равенства в нижней строке правого столбца таблицы:  $2 = 8 - 6 \cdot 1$ . Далее вместо цифры 6 подставляется ее значение из равенства строкой выше:  $2 = 8 - (46 - 8 \cdot 5) \cdot 1$  и т. д. Полная цепочка подстановок и преобразований выглядит так:

$$\begin{aligned} 2 &= 8 - (46 - 8 \cdot 5) \cdot 1 = 8 - 46 + 8 \cdot 5 = 8 \cdot 6 - 46 = (54 - 46) \cdot 6 - 46 = 54 \cdot 6 - 46 \cdot 6 - 46 = 54 \cdot 6 - \\ &- 46 \cdot 7 = 54 \cdot 6 - (1234 - 54 \cdot 22) \cdot 7 = 54 \cdot 6 - 1234 \cdot 7 + 54 \cdot 154 = \mathbf{54 \cdot 160 + (-7) \cdot 1234} = 8640 - \\ &- 8638. \end{aligned}$$

Из выражения перед последним знаком равенства (выделено) следует, что для нашего примера  $u = -7$  и  $v = 160$  в соответствии с формой записи в (1.4).

Исследованием целых чисел занимался швейцарский математик Леонард Эйлер (Leonard Euler). Один из важных вопросов его исследования: сколько существует натуральных чисел, не превосходящих некоторое число  $n$  и взаимно простых с  $n$ ? Ответ на этот вопрос связан с каноническим разложением числа  $n$  на простые множители (см. выше основную теорему арифметики и пример 4). Так, если



$$n = p_1^{a_1} \times p_2^{a_2} \times \dots \times p_n^{a_n} \quad (1.5)$$

где  $p_1, p_2, \dots, p_n$  – разные простые множители, то число  $\varphi(n)$  натуральных чисел, не превосходящих  $n$  и взаимно простых с  $n$  можно точно определить по формуле

$$\varphi(n) = n \times \left(1 - \frac{1}{p_1}\right) \times \left(1 - \frac{1}{p_2}\right) \times \dots \times \left(1 - \frac{1}{p_n}\right) \quad (1.6)$$

Число натуральных чисел, не превосходящих  $n$  и, взаимно простых с  $n$ , называется *функцией Эйлера* и обозначается  $\varphi(n)$ .

**Пример 16.** Определить количество натуральных чисел, не превосходящих 12 и взаимно простых с 12.

Взаимно простыми с 12 будут четыре числа 1, 5, 7, 11, т. е.  $\varphi(12) = 4$  – получено методом «ручного» подсчета.

Теперь подсчитаем  $\varphi(12)$  по (1.5). Вспомнив примеры 4 и 5, запишем каноническое разложение числа 12:  $12 = 2 \cdot 2 \cdot 3 = 2^2 \cdot 3$ , т. е.  $p_1 = 2$ ,  $p_2 = 3$ . Теперь подсчитаем функцию Эйлера,  $\varphi(12)$ :

$$\varphi(12) = 12 \cdot (1 - 1/2) \cdot (1 - 1/3) = 4.$$

Если  $p$  – простое число, то  $\varphi(p) = p - 1$ , если числа  $p$  и  $q$  являются простыми и  $p \neq q$ , то

$$\varphi(p) = (p - 1) (q - 1). \quad (1.7)$$

#### 1.1.4 Модулярная арифметика и обратные числа по модулю

Понятие «модулярная арифметика» ввел немецкий ученый Гаусс. В этой арифметике мы интересуемся остатком от деления числа  $a$  на число  $n$  ( $n$  – натуральное число и  $n > 1$ ). Если таким остатком является число  $b$ , то можно записать:

$$a \equiv b \pmod{n} \text{ или } a \equiv b \pmod{n}.$$

Такая формальная запись читается как « $a$  сравнимо с  $b$  по модулю  $n$ ».

При целочисленном (в том числе и нулевом) результате деления числа  $a$  на число  $n$  справедливо  $a = b + kn$ .

**Пример 17.** При  $a=13$  и  $n=4$  имеем  $b=1$ , т.е.  $13 = 1 + 3 \cdot 4$ . Для данного примера справедлив вывод: число 13 по модулю 4 равно 1 или числа 13 и 1 равны по модулю 4.

**Пример 18.** Справедливы следующие сравнения чисел:

$$-5 \equiv 7 \pmod{4} \equiv 11 \pmod{4} \equiv 23 \pmod{4} \equiv 3 \pmod{4}.$$

Иногда  $b$  называют *вычетом* по модулю  $n$ , а числа  $a$  и  $b$  называют *сравнениями* (по модулю  $n$ ).

Модулярная арифметика так же коммутативна, ассоциативна и дистрибутивна, как и обычная арифметика. В силу этих свойств сравнения можно почленно складывать, вычитать, умножать, возводить в степень ( $a^m \bmod n \equiv (a \bmod n)^m$ , если  $a \equiv b \bmod n$ , то  $a^m \equiv b^m \bmod n$ ); другие примеры см. в [2]).

Указанные свойства позволяют упрощать сложность и время выполнения многих вычислений. Криптография использует множество вычислений по модулю  $n$ , потому что задачи типа вычисления *дискретных логарифмов* и квадратных корней очень трудны. Кроме того, с вычислениями по модулю удобнее работать, потому что они ограничивают диапазон всех промежуточных величин и результата.

**Обратные числа в модулярной арифметике.** Традиционно: обратное к 7 равно  $7^{-1}=1/7$ , так как  $7 \cdot (1/7) = 1$ . В модулярной арифметике запись уравнения в виде

$$ax \equiv 1 \bmod n \quad (1.8)$$

предусматривает поиск таких значений  $x$  и  $k$ , которые удовлетворяют равенству

$$ax = nk + 1. \quad (1.9)$$

Общая задача решения уравнения (1.8) может быть сформулирована следующим образом: найти такое  $x$ , что

$$1 \equiv ax \bmod n. \quad (1.10)$$

Уравнение (1.10) имеет единственное решение, если  $a$  и  $n$  – взаимно простые числа, в противном случае – решений нет.

Уравнение (1.10) можно переписать в ином виде:

$$a^{-1} \equiv x \bmod n. \quad (1.11)$$

**Пример 19.** При  $a=5$  и  $n=14$  получим  $x=3$ :  $5^{-1} \equiv 3 \bmod 14$ , так как  $5 \cdot 3 \bmod 14 \equiv 1$ .

Если  $\text{НОД}(a, n) = 1$ , то  $a^{-1}a \equiv 1 \bmod n$ ,  $a^{-1}$  – число, обратное  $a$  по модулю  $n$ .

Справедливо также: если

$$x^{-1} \equiv y \bmod n, \text{ то } y^{-1} \equiv x \bmod n. \quad (1.12)$$

В силу приведенных рассуждений и обоснований (1.12) удовлетворяют такие числа, при которых выполняется равенств

$$xy + kn = 1, \quad (1.13)$$

где  $k$  – целое число (результат деления  $xy/n$ ).

Нахождение чисел, обратных по модулю, легко реализуется с помощью расширенного алгоритма Евклида (см пример 16 и программную реализацию алгоритма в [2]).

Пример 20. Решить уравнение  $7y \equiv 1 \pmod{40}$  или  $y^{-1} \equiv 7 \pmod{40}$

Находим НОД (7,40) – прямая прогонка (алгоритм Евклида):

$$40 = 7 \cdot 5 + 5,$$

$$7 = 5 \cdot 1 + 2,$$

$$5 = 2 \cdot 2 + 1, \text{ т. е. НОД } (7,40) = 1.$$

Обратная подстановка (приведение (1.12) к форме (1.13):

$1 = 5 - 2 \cdot 2 = 5 - 2(7 - 5 \cdot 1) = 5 \cdot 3 + 7(-2) = (40 - 7 \cdot 5)3 + 7(-2) = 40 \cdot 3 + 7(-17) = kn + xy = 1 \pmod{n}$ , или  $7(-17) = 7y$ , так как  $-17 \pmod{40} = 23$ , то  $y=23$ : число 23 является обратным числу 7 по модулю 40.

**Малая теорема Ферма.** Если  $n$  – простое число, а число  $a$  не кратно  $n$ , то справедливо:

$$a^n \equiv 1 \pmod{n}. \quad (1.14)$$

В соответствии с *обобщением Эйлера* приведенной теоремы, если НОД ( $a, n$ ) = 1, то справедливо:

$$a^{\varphi(n)} \pmod{n} \equiv 1. \quad (1.15)$$

Последнее выражение можно переписать в следующем виде:

$$a^{-1} \pmod{n} \equiv a^{\varphi(n)-1} \pmod{n}. \quad (1.16)$$

Пример 21. Найти число, обратное 5 по модулю 7. Число 7 является простым. Поэтому  $\varphi(7) = 7 - 1 = 6$ . Теперь с помощью (1.16) получаем:  $5^{6-1} \pmod{7} \equiv 5^5 \pmod{7} \equiv 3$ .

Таким образом,  $5^{-1} \pmod{7} \equiv 3$  или  $5 \cdot 3 \equiv 1 \pmod{7}$ .

## 1.2 ПРАКТИЧЕСКОЕ ЗАДАНИЕ

*Рекомендация!* Перед выполнением практического задания можно познакомиться с функционалом программного средства *L\_PROST*, особенностями выполнения с его помощью операций над простыми числами и являющегося приложением на компакт-диске к [5].

На рисунке 1.1 представлено основное диалоговое окно программы после запуска исполнительного файла *L\_PROST.EXE*. Как видно, средство позволяет генерировать простые числа, осуществлять проверку чисел на простоту и определять простые числа в заданном интервале.

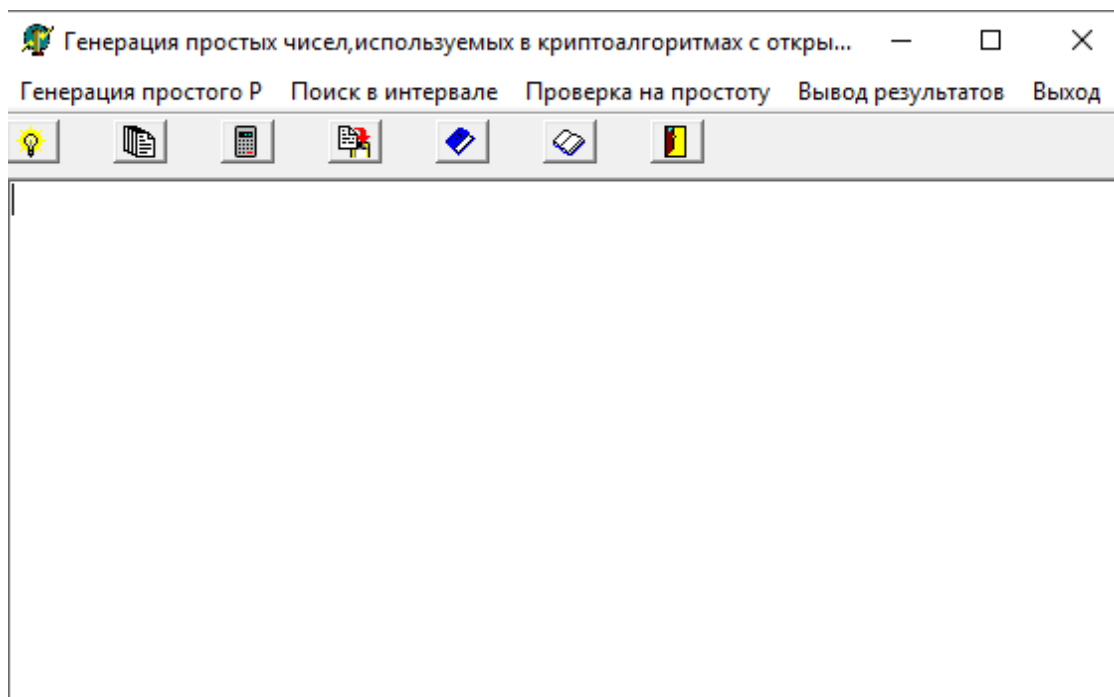


Рисунок 1.1 Основное диалоговое окно программного средства *L\_PROST*

1. Используя *L\_PROST*, найти все простые числа в интервале  $[2, n]$ . Значение  $n$  соответствует варианту из таблицы 1.1, указанному преподавателем.

Таблица 1.1 Варианты задания

Вариант	$m$	$n$
1	450	503
2	521	553
3	367	401
4	421	457
5	499	531
6	431	471
7	540	577
8	667	703
9	399	433
10	587	621
11	555	591
12	354	397
13	379	411
14	632	663
15	447	477

Подсчитать количество простых чисел в указанном интервале. Сравнить это число с  $n/\ln(n)$  (см. выше пример 15).

2. Повторить п.1 для интервала  $[m, n]$ .

Сравнить полученные результаты с «ручными» вычислениями, используя «решето Эратосфена» (см. примеры 11 и 12).

3. Записать числа  $m$  и  $n$  в виде произведения простых множителей (форма записи – каноническая).

4. Проверить, является ли число, состоящее из конкатенации цифр  $m||n$  (таблица 1.1), простым.

5. Найти НОД ( $m, n$ ).

### **Основное задание.**

6. Разработать авторское приложение в соответствии с целью лабораторной работы. Приложение должно реализовывать следующие операции:

- вычислять НОД двух либо трех чисел;
- выполнять поиск простых чисел.

7. С помощью созданного приложения выполнить задания по условиям п.п. 1 и 2.

8. Результаты выполнения работы оформить в виде отчета по установленным правилам.

### **ВОПРОСЫ ДЛЯ КОНТРОЛЯ И САМОКОНТРОЛЯ**

1. Дать определение понятий: целое число, натуральное число, делимость чисел, собственный делитель, НОД.

2. Сформулировать основную теорему арифметики. Представить примеры ее применения.

3. Пояснить сущность проблемы факторизации и ее связь с прикладной криптографией.

4. Найти НОД:

пар чисел: 333 и 100; 56 и 200; 99 и 200; 61 и 987; 123 и 456;

трех чисел: 21, 43, 342; 57, 31, 200; 42, 11, 98.

5. Записать каноническое разложение чисел: 2770, 3780, 6224.

6. Записать соотношение Безу. Показать пример его практического использования.

7. Подсчитать число взаимно простых чисел с числами 2770, 3780, 6224.

8. Сформулировать малую теорему Ферма. Показать примеры ее практического применения.

9. Сформулировать основные свойства модулярной арифметики.

10. Пояснить порядок операций на основе расширенного алгоритма Евклида.

11. Найти числа обратные к  $a$  по модулю  $n$ :  $a = 41, n = 143$ ;  $a = 13, n = 71$ .

## Лабораторная работа № 2

### Исследование криптографических шифров на основе подстановки (замены) символов

**Цель:** изучение и приобретение практических навыков разработки и использования приложений для реализации подстановочных шифров.

**Задачи:**

1. Закрепить теоретические знания по алгебраическому описанию, алгоритмам реализации операций зашифрования/расшифрования и оценке криптостойкости подстановочных шифров.
2. Ознакомиться с особенностями реализации и свойствами различных подстановочных шифров на основе готового программного средства (L\_LUX).
3. Разработать приложение для реализации указанных преподавателем методов подстановочного зашифрования/расшифрования.
4. Выполнить исследование криптостойкости шифров на основе статистических данных о частотах появления символов в исходном и зашифрованном сообщениях.
5. Оценить скорость зашифрования/расшифрования реализованных способов шифров.
6. Результаты выполнения лабораторной работы оформить в виде описания разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента.

#### 2.1 ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Лабораторная работа является первой в цикле работ, относящихся к криптографическим шифрам. Основные теоретические сведения и определения из данной предметной области можно найти в [2].

В этом разделе материалов кратко рассмотрим лишь сведения, имеющие непосредственное отношение к цели и задачам лабораторной работы.

Сущность подстановочного шифрования состоит в том, что, исходный текст (из множества  $M$ ) и зашифрованный текст (из множества  $C$ ) основаны на использовании одного и того же или разных алфавитов, а тайной или ключевой информацией является алгоритм подстановки.

Если исходить из того, что используемые алфавиты являются конечными множествами, то в общем случае каждой букве  $a_x$  алфавита  $A_M$  ( $a_x \in A_M$ ) для создания сообщения  $M_i$  ( $M_i \in M$ ) соответствует буква  $a_y$  или множество

букв  $\{A_{xC}\}$  для создания шифртекста  $C_i$  ( $C_i \in C$ ). Важно, чтобы во втором случае любые два множества (например,  $\{A_{xC}\}_b$  и  $\{A_{xC}\}_n$ ,  $b \neq n$ ,  $1 \leq b, n, x, y \leq N$ ,  $N$  – мощность алфавита), используемые для замены разных букв открытого текста, не пересекались:

$$\{A_{xC}\}_b \cap \{A_{xC}\}_n = 0.$$

Если в сообщении  $M_i$  содержится несколько букв  $a_x$ , то каждая из них заменяется на символ  $a_y$  либо на любой из символов  $\{A_{xC}\}$ . За счет этого с помощью одного ключа можно сгенерировать различные  $C_i$  для одного и того же  $M_i$ . Так как множества  $\{A_{xC}\}_b$  и  $\{A_{xC}\}_n$  попарно не пересекаются, то по каждому символу  $C_i$  можно однозначно определить, какому множеству он принадлежит, и, следовательно, какую букву открытого сообщения  $M_i$  он заменяет. В силу этого открытое сообщение восстанавливается из зашифрованного однозначно.

Приведенные утверждения справедливы для следующих типов подстановочных шифров:

- *моноалфавитных* (шифры однозначной замены или простые подстановочные),
- *полиграммных*,
- *омофонических* (однозвучные шифры или шифры многозначной замены),
- *полиалфавитных*.

Кратко поясним особенности указанных шифров.

### ***2.1.1. Моноалфавитные шифры подстановки***

В данных шифрах операция замена производится только над каждым отдельным символом сообщения  $M_i$ . Для наглядной демонстрации шифра простой замены достаточно выписать под заданным алфавитом тот же алфавит, но в другом порядке или, например, со смещением. Записанный таким образом алфавит называют алфавитом замены.

Максимальное количество ключей для любого шифра этого вида не превышает  $N!$ , где  $N$  – количество символов в алфавите.

Для математического описания криптографического преобразования предполагаем, что зашифрованная буква  $a_y$  ( $a_y \in C_i$ ), соответствующая символу  $a_x$  ( $a_x \in M_i$ ), находится на позиции

$$y \equiv x + k \pmod{N}, \quad (2.1)$$

где  $x, y$  – индекс (порядковый номер, начиная с 0) символа в используемом алфавите,  $k$  – ключ.

Для расшифрования сообщения  $C_i$  необходимо произвести расчеты обратные (2.1), т. е.:

$$x \equiv y - k \pmod{N}. \quad (2.2)$$

Соотношения (2.1) и (2.2) соответствует классический шифр подстановки: **шифр Цезаря**. Согласно описаниям историка Светония в книге «Жизнь двенадцати цезарей» данный шифр использовался Гаем Юлием Цезарем для секретной переписки со своими генералами (I век до н.э.) [6] (в этой книге любознательный читатель найдет также много исторической информации по криптографии).

Пример 1. Имеем открытый текст  $M_i = \langle cba \rangle$ . На основе шифра Цезаря  $C_i = \langle fed \rangle$ .

Здесь  $k = 3$ ,  $N = 26$ . Первый символ открытого текста ( $c$ ) имеет индекс 2 (помним, что начальный символ алфавита ( $a$ ) имеет нулевой индекс). Значит, первый символ шифртекста ( $e$ ) будет иметь индекс  $2 + k = 5$ . А такой индекс в алфавите принадлежит символу  $f$  и т.д.

Известное послание Цезаря *VENI VIDI VICI* (в переводе на русский означает «Пришел, Увидел, Победил»), направленное его другу Аминтию после победы над понтийским царем Фарнаком, выглядело бы в зашифрованном виде так: *YNQL YLGL YLFL*.

Применительно к русскому языку суть его состоит в следующем. Выписывается исходный алфавит (А, Б, ..., Я), затем под ним выписывается тот же алфавит, но с циклическим сдвигом на 3 позиции влево.

Существуют различные модификации шифра Цезаря, в частности, *Атбаш* и *лозунговый шифр*.

**Атбаш.** В Ветхом Завете существует несколько фрагментов из священных текстов, которые зашифрованы с помощью шифра замены, называемого Атбаш. Этот шифр состоит в замене каждой буквы другой буквой, которая находится в алфавите на таком же расстоянии от конца алфавита, как оригинальная буква – от начала. Например, в русском алфавите буква А заменяется на Я, буква Б – на Ю и т.д. В оригинальном Ветхом Завете использовались буквы еврейского алфавита. Так, в книге пророка Иеремии слово «Бабель» (Вавилон) зашифровано как «Шешах» [6].

Одним из существенных недостатков моноалфавитных шифров является их низкая криптостойкость. Зачастую метод криптоанализа базируется на частоте встречаемости букв исходного текста.

Если в открытом сообщении часто встречается какая-либо буква, то в шифрованном сообщении также часто будет встречаться соответствующий ей символ. Еще в 1412 г. Шихаба ал-Калкашанди в своем труде «Субх ал-Ааша» привел таблицу частоты появления арабских букв в тексте на основе анализа



текста Корана. Для разных языков мира существуют подобные таблицы. Так, например, для букв русского алфавита по данным «Национального корпуса русского языка» [7] (Корпус — это информационно-справочная система, основанная на собрании текстов на некотором языке в электронной форме. Национальный корпус представляет данный язык на определенном этапе (или этапах) его существования и во всём многообразии жанров, стилей, территориальных и социальных вариантов и т. п.) такая таблица выглядит следующим образом (таблица 2.1).

Таблица 2.1. Частота появления букв русского языка в текстах

№ п/п	Буква	Частота, %	№ п/п	Буква	Частота, %
1	О	10.97	18	Ь	1.74
2	Е	8.45	19	Г	1.70
3	А	8.01	20	З	1.65
4	И	7.35	21	Б	1.59
5	Н	6.70	22	Ч	1.44
6	Т	6.26	23	Й	1.21
7	С	5.47	24	Х	0.97
8	Р	4.73	25	Ж	0.94
9	В	4.54	26	Ш	0.73
10	Л	4.40	27	Ю	0.64
11	К	3.49	28	Ц	0.48
12	М	3.21	29	Щ	0.36
13	Д	2.98	30	Э	0.32
14	П	2.81	31	Ф	0.26
15	У	2.62	32	Ъ	0.04
16	Я	2.01	33	Ё	0.04
17	Ы	1.90			

Существуют подобные таблицы для пар букв (биграмм). Например, часто встречаемыми биграммами являются «то», «но», «ст», «по», «ен» и т.д. Другой прием взлома шифров основан на исключении возможных сочетаний букв. Например, в текстах (если они написаны без орфографических ошибок) нельзя встретить сочетания «чя», «щы», «ьъ» и т.п. Таблицы с частотами (вероятностями) встречаемости пар и большего числа буквосочетаний существуют для разных алфавитов. Пример использования частотных свойств символов алфавита английского языка для шифроанализа можно найти на страницах 17-19 пособия [8].

Расшифрование сообщения производится по правилу, которое мы рассматривали на выше проанализированных примерах.

Применяя одновременно операции сложения и умножения по модулю  $n$  над элементами множества (индексами букв алфавита), можно получить *систему подстановок*, которую называют **аффинной системой подстановок Цезаря**. Определим процедуру зашифрования в такой системе:

$$y \equiv ax + b \pmod{N}, \quad (2.3)$$

где  $a$  и  $b$  – целые числа.

При этом взаимно однозначные соответствия между открытым текстом и шифртекстом будут иметь место только при выполнении следующих условий:  $0 \leq a, b < N$ , наибольший общий делитель (НОД) чисел  $a, b$  равен 1, т.е. эти числа являются *взаимно простыми*.

Приме 4. Пусть  $N=26$ ,  $a=3$ ,  $b=5$ . Тогда,  $\text{НОД}(3, 26) = 1$ , и мы получаем следующее соответствие между индексами букв:

$x$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
$3x+5$	5	8	11	14	17	20	23	0	3	6	9	12	15	18	21	24	1	4	7	10	13	16	19	22	25	2

Преобразуя числа в буквы английского алфавита, получаем следующее соответствие для букв открытого текста и шифртекста:

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z  
F I L O R U X A D G J M P S V Y B E H K N Q T W Z C

Если  $M_i = \text{'VENIVIDIVI'}$ , то получение зашифрованного сообщения в деталях показывает таблица 2.2.

Таблица 2.2. Иллюстрация получения шифртекста на основе аффинной системы подстановок Цезаря

$M_i$	V	E	N	I	V	I	D	I	V	I	C	I
$x$	21	4	13	8	21	8	3	8	21	8	2	8
$y=3x+5$	16	17	18	3	16	3	14	3	16	3	11	3
$C_i$	Q	R	S	C	Q	C	O	C	Q	C	L	C

Таким образом, зашифрованное сообщение будет таким:  $C_i = \text{'QRSCQCOCQCLC'}$ .

Расшифрование основано на использовании соотношения

$$x \equiv a^{-1}(y+N-b) \pmod{N}, \quad (2.4)$$

где  $a^{-1}$  – обратное к  $a$  число по модулю  $N$ , т. е. оно удовлетворяет уравнению  $aa^{-1} \equiv 1 \pmod{N}$ .

### 2.1.2. Полиграммные шифры

В таких шифрах одна подстановка соответствует сразу нескольким символам исходного текста.

Первым известным шифром этого типа является *шифр Порты* [9]. Шифр представляется в виде таблицы. Наверху горизонтально и слева вертикально записывался стандартный алфавит. В ячейках таблицы записываются числа в определенном порядке. Одним из возможных вариантов такой таблицы для алфавита русского языка будет показанная ниже таблица, точнее – ее фрагмент (таблица 2.3).

Таблица 2.3. Фрагмент шифра Порты для алфавита русского языка, состоящего из 33 букв

	А	Б	В		Э	Ю	Я
А	001	002	003		031	032	033
Б	034	035	036		064	065	066
В	067	068	069		097	098	099
Г	100	101	102		130	131	132
Ю	1024	1025	1026		1054	1055	1056
Я	1057	1058	1059		1087	1088	1089

Шифрование выполняется парами букв исходного сообщения. Первая буква пары указывает на строку, вторая – на столбец. В случае нечетного количества букв в сообщении  $M_i$  к нему добавляется вспомогательный символ, например, «А».

*Пример 5.* Исходное сообщение  $M_i = \text{'АВВА'}$ . Сообщение состоит из двух пар (биграмм): АВ и ВА и будет зашифровано так: 003067.

Другими известными полиграммными шифрами являются *шифр Плейфера* и *шифр Хилла* [9].

С точки зрения криптостойкости рассматриваемый тип шифров имеет преимущества перед моноалфавитными шифрами. Это связано с тем, что распределение частот групп букв значительно более равномерное, чем отдельных символов. Во-вторых, для эффективного частотного анализа требуется больший размер зашифрованного текста, так как число различных групп букв значительно больше, чем мощность алфавита.

### 2.1.3 Омофонические шифры

*Омофонические шифры (омофоническая замена) или однозвучные шифры подстановки* создавались с целью увеличить сложность частотного анализа шифртекстов путем маскировки реальных частот появления символов текста с помощью *омофонии*.

В 1401 г. Симеоне де Крема стал использовать таблицы омофонов для сокрытия частоты появления гласных букв в тексте при помощи более чем одной подстановки. Такие шифры позже стали называться *шифрами многозначной замены* или *омофонами* (омофоны – от греч. *homos* – одинаковый и *phone* – звук) – слова, которые звучат одинаково, но пишутся по-разному и имеют разное значение; очень много подобных слов содержит английский язык). Они получили развитие в XV веке. В книге «Трактат о шифрах» Леона Баттисты Альберти (итальянский ученый, архитектор, теоретик искусства, секретарь папы Климентия XII), опубликованной в 1466 г. [10], приводится описание шифра замены, в котором каждой букве ставится в соответствие несколько эквивалентов, число которых пропорционально частоте встречаемости буквы в открытом тексте,  $M_i$ . В этих шифрах буквы исходного алфавита соответствуют более чем одному символу из алфавита замены. Обычно символам исходного текста с наивысшей частотой дают большее количество эквивалентов, чем более редким символам. Таким образом, распределение частоты становится более равномерным, сильно затрудняя частотный анализ.

В таблице 2.4 представлен фрагмент таблицы подстановок для алфавита русского языка [11].

При шифровании символ исходного сообщения заменяется на любую подстановку из «своего» столбца. Если символ встречается повторно, то, как правило, используют разные подстановки. Например, исходное сообщение «АБРАМОВ» после зашифрования может выглядеть так: «357 990 374 678 037 828 175» [11].

**Книжный шифр.** Заметным вкладом греческого ученого Энея Тактика в криптографию является предложенный им так называемый книжный шифр [10, 11]. После Первой мировой войны книжный шифр приобрел иной вид. Шифрозамена для каждой буквы определялась набором цифр, которые указывали на номер страницы, строки и позиции в строке (вспомните пример использования такого шифра известными героями фильма «17 мгновений весны»). Даже с формальной стороны отсутствие полной электронной базы изданных к настоящему времени книг делает процедуру взлома шифра практически не выполнимой.

Таблица 2.4 Фрагмент таблицы подстановок для системы омофонов

№ п/п	А	Б	В	...	М	...	О	...	Р	...	Я
1	311	128	175	...	037	...	248	...	064	...	266
2	357	950	194	...	149	...	267	...	189	...	333
...	...	...	...	...	...	...	...	...	...	...	...
16	495	990	199	...	349	...	303	...	374	...	749
...	...		...	...	...	...	...	...	...	...	...
20	519		427	...	760	...	306	...	469	...	845
...	...		...	...	...	...	...	...	...		
32	637		524	...	777	...	432	...	554		
...	...		...				...	...	...		
45	678		644				824	...	721		
...	...						...	...	...		
47	776						828	...	954		
...	...						...				
80	901						886				
...							...				
110							903				

#### 2.1.4. Полиалфавитные шифры

*Полиалфавитные* (или *многоалфавитные*) шифры состоят из нескольких шифров однозначной замены. Выбор варианта алфавита для зашифрования одного символа зависит от особенностей метода шифрования.

**Диск Альберти.** В «Трактате о шифрах» [10] Альберти приводит первое точное описание *многоалфавитного шифра* на основе *шифровального диска* (см. рис. 2.1).

Он состоял из двух дисков – внешнего неподвижного и внутреннего подвижного дисков, на которые были нанесены буквы алфавита. Процесс шифрования заключался в нахождении буквы открытого текста на внешнем диске и замене ее на букву с внутреннего диска, стоящую под ней. После этого внутренний диск сдвигался на одну позицию и шифрование второй буквы производилось уже по-новому шифралфавиту. Ключом данного шифра являлся порядок расположения букв на дисках и начальное положение внутреннего диска относительно внешнего.



Рисунок 2.1. Реплика диска Альберти, используемого Конфедерацией во время Гражданской войны в Америке [12]

**Таблица Трисемуса.** В 1518 году в развитии криптографии был сделан важный шаг благодаря появлению в Германии первой печатной книги по криптографии. Аббат Иоганнес Трисемус, настоятель монастыря в Вюрцбурге, написал книгу «Полиграфия», в которой он описал ряда шифров, один из которых развивает идею *многоалфавитной подстановки*. Зашифрование осуществляется так: заготавливается *таблица подстановки* (так называемая «*таблица Трисемуса*» – таблица со стороной равной  $N$ , где  $N$  – мощность алфавита), где первая строка – это алфавит, вторая – алфавит, сдвинутый на один символ, и т. д. При зашифровании первая буква открытого текста заменяется на букву, стоящую в первой строке, вторая – на букву, стоящую во второй строке, и т. д. После использования последней строки вновь возвращаются к первой.

**Пример.** Рассмотрим процесс зашифрования сообщения  $M_i = \langle \text{БГТУ} \rangle$ , используя таблицу, фрагмент которой показан на рисунке 2.2.

А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ
Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы
В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь
Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э
Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю
Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я

Рисунок 2.2 Фрагмент таблицы Трисемуса для алфавита русского языка

Стрелками на приведенном рисунке показан принцип зашифрования каждого символа открытого текста. Из этого следует, что шифртекст имеет вид:  $C_i = \langle \text{БДФЦ} \rangle$ .

В указанной книге Трисемус впервые систематически описал применение шифрующих таблиц, заполненных алфавитом в случайном порядке. Для получения такого шифра подстановки обычно использовались таблица для записи букв алфавита и *ключевое слово* (или фраза). Можно найти определенную аналогию с системой шифрования Цезаря с ключевым словом. В таблицу сначала вписывалось по стрелкам ключевое слово, причем повторяющиеся буквы также отбрасывались. Затем эта таблица дополнялась не вошедшими в нее буквами алфавита по порядку.

Таким образом, ключом в таблицах Трисемуса является ключевое слово и размер таблицы. При шифровании буква открытого текста заменяется буквой, расположенной ниже нее в том же столбце. Если буква текста оказывается в нижней строке таблицы, тогда для шифртекста берут самую верхнюю букву из того же столбца.

Указанный размер таблицы для алфавита русского языка может соответствовать 4x8 либо 8x4.

Пример 6. Пусть  $M_i = \langle \text{ПРИШЕЛУВИДЕЛПОБЕДИЛ} \rangle$ , а ключевое слово –  $\langle \text{ЦЕЗАРЬ} \rangle$ . Используем таблицу 8x4.

Ц	Е (Ё)	З	А
Р	Ь	Б	В
Г	Д	Ж	И
Й	К	Л	М
Н	О	П	С
Т	У	Ф	Х
Ч	Ш	Щ	Ъ
Ы	Э	Ю	Я

Следуя вышеуказанного принципу подстановки, получим  $C_i = \langle \text{ФГМЭЫПШИМКЫПФУЖЪКМП} \rangle$ .

**Шифр Виженера.** В 1586 г. французский дипломат Блез Виженер представил перед комиссией Генриха III описание простого, но довольно стойкого шифра, в основе которого лежит таблица Трисемуса.

В этом шифре мы имеем дело с последовательностью сдвигов, циклически повторяющейся. Основная идея заключается в следующем. Создается таблица (таблица Виженера) размером  $N \cdot N$  ( $N$  – число знаков в используемом алфавите). Эти знаки могут включать не только буквы, но и, например, пробел или



иные знаки. В первой строке таблицы записывается весь используемый алфавит. Каждая последующая строка получается из предыдущего циклического сдвига последней на 1 символ влево. Таким образом, при мощности алфавита (английского языка) равной 26, необходимо выполнить последовательно 25 сдвигов для формирования всей таблицы.

Более подробное описание шифра можно найти, например, в [2] (с. 41–43).

Листинг 2.1 содержит часть кода, реализующего алгоритм шифрования Виженера.

Следует добавить, что в 1863 г. Фридрих Касиски опубликовал алгоритм атаки на этот шифр, хотя известны случаи его взлома шифра некоторыми опытными криптоаналитиками и ранее. В частности, в 1854 г. шифр был взломан изобретателем первой аналитической вычислительной машины Чарльзом Бэббиджем. Этот факт стал известен только в XX в., когда группа ученых разбирала вычисления и личные заметки Бэббиджа [5]. Несмотря на это шифр Виженера имел репутацию исключительно стойкого к «ручному» взлому еще долгое время. Так, известный писатель и математик Чарльз Доджсон (Льюис Кэрролл) в своей статье «Алфавитный шифр», опубликованной в детском журнале в 1868 г., назвал шифр Виженера невзламываемым. В 1917 г. научно-популярный журнал «Scientific American» также отозвался о шифре Виженера, как о неподдающемся взлому [13].

```
/*
 * главный цикл, который проходит по входной строке
 */
foreach (char symbol in input)
{
    /* characters - алфавит,
       keyword - ключевое слово,
       N - мощность алфавита,
       keyword_index - индекс текущей буквы ключевого слова
    */
    int c = (Array.IndexOf(characters, symbol) +
             Array.IndexOf(characters, keyword[keyword_index])) % N;

    /* result - результирующая строка */
    result += characters[c];

    keyword_index++;

    /* циклический проход по ключевому слову */
    if ((keyword_index + 1) == keyword.Length)
        keyword_index = 0;
}
```

Листинг 2.1 Фрагмент кода, реализующего алгоритм шифра Виженера

*Роторные машины.* Идеи Альберти и Виженера использовались при создании электромеханических роторных машин первой половины XX века. Некоторые из них использовались в разных странах вплоть до 1980-х годов. В большинстве из них использовались роторы (механические колеса), взаимное расположение которых определяло текущий алфавит шифрозамен, используемый для выполнения подстановки. Наиболее известной из роторных машин является немецкая машина времен Второй мировой войны «Энигма». Более детально изучению и практическому анализу «Энигмы» далее будет посвящена отдельная лабораторная работа.

К полиалфавитным относится также шифр на основе «одноразового блокнота».

Много полезной информации по рассмотренному классу шифров можно найти в [14].

Существует определенное сходство между подстановочными шифрами и *шифрами на основе гаммирования*. Последние рассматриваются как самостоятельный класс. Такие шифры схожи с подстановочными (и в определенном плане – с *перестановочными*) тем, что в обоих случаях можно использовать табличное представление выполняемых операций на основе ключей. В шифрах на основе гаммирования и в подстановочных шифрах при зашифровании происходит подмена одних символов на другие.

Гаммирование будем рассматривать и изучать более подробно далее в лабораторной работе № xx.

## 2.2. ПРАКТИЧЕСКОЕ ЗАДАНИЕ

*Рекомендация!* Перед выполнением практического задания можно познакомиться с особенностями работы программного средства *L\_LUX*, реализующего подстановочные (и другие) методы зашифрования/расшифрования текстовой информации и являющегося приложением на компакт-диске к [5].

На рисунке 2.3 представлено основное диалоговое окно программы после запуска исполнительного файла *L\_LUX.EXE*.

Программа понятна и проста в использовании с точки зрения интерфейса и функционала. Основная часть окна – текстовый редактор, в котором можно набирать текст либо размещать скопированный фрагмент из другого текстового документа (вставка – Ctrl+V). Здесь же отображается зашифрованный текст, а также сформированные программой распределения частот (в виде гистограмм) для исходного и зашифрованного текстов.

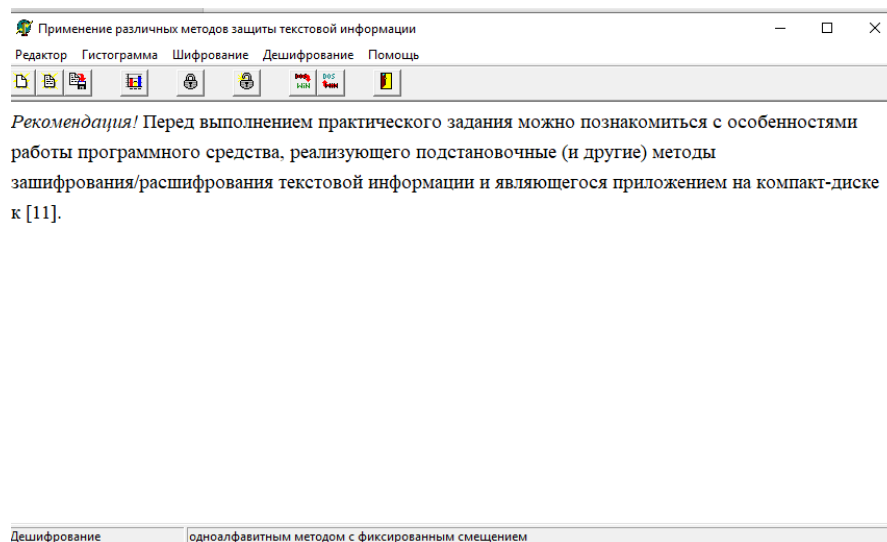


Рисунок 2.3 Основное диалоговое окно программного средства *L\_LUX*

На рисунке 2.4 для примера и сравнения приведены гистограммы для исходного (в окне редактора на рис. 2.3) и зашифрованного документов (обратим, внимание, что отдельные буквы – строчные и прописные – рассматриваются здесь как разные, что не соответствует традиционному подходу).

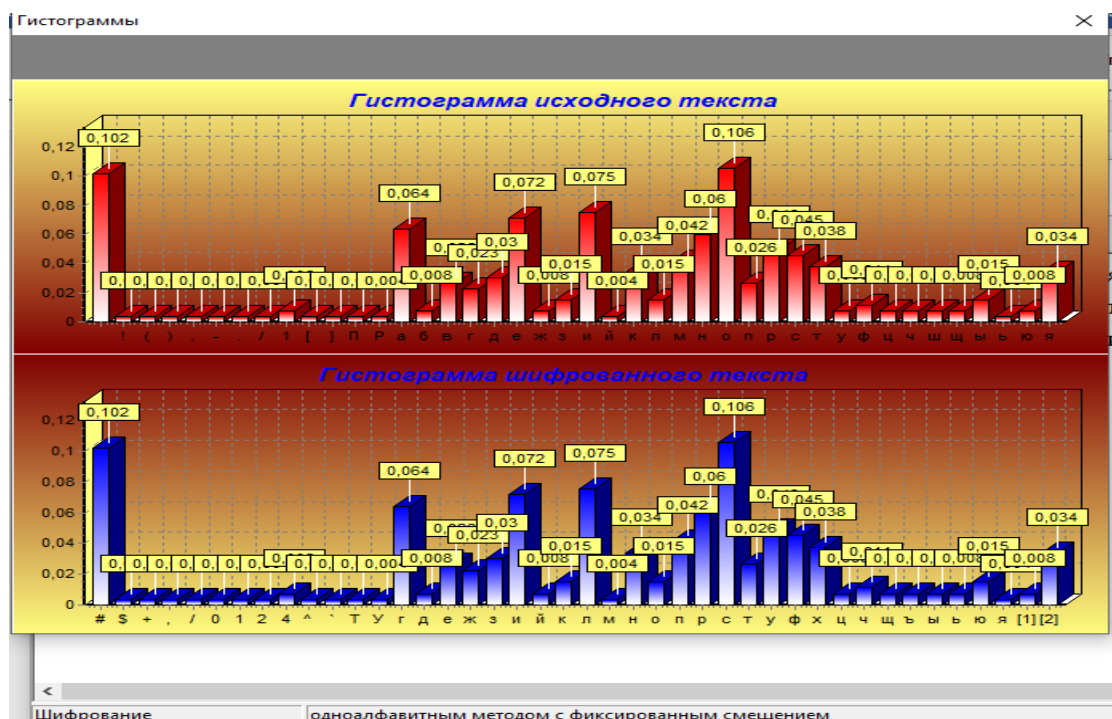


Рисунок 2.4 Гистограммы для исходного и зашифрованного текстовых документов

### Основное задание.

1. Разработать авторское приложение в соответствии с целью лабораторной работы. Приложение должно реализовывать следующие операции:

- выполнять зашифрование/расшифрование текстовых документов (объемом не менее 5 тысяч знаков) созданных на основе алфавита языка в соответствии с нижеследующей таблицей вариантов задания; при этом следует использовать шифры подстановки из третьего столбца данной таблицы;

#### Варианты задания

Вариант	алфавит	шифр
1	белорусский	1. На основе соотношений (2.1) и (2.2); $k=5$ 2. Виженера, ключевое слово – собственная фамилия
2	русский	1. На основе аффинной системы подстановок Цезаря; $a=7$ , $b=10$ 2. Виженера, ключевое слово – собственная фамилия
3	английский	1. Шифр Цезаря с ключевым словом, ключевое слово – собственная фамилия 2. Таблица Трисемуса, ключевое слово – собственное имя
4	немецкий	1. На основе соотношений (2.1) и (2.2); $k=7$ 2. Таблица Трисемуса, ключевое слово – enigma
5	польский	1. На основе аффинной системы подстановок Цезаря; $a=5$ , $b=7$ 2. Шифр Порты
6	белорусский	1. На основе соотношений (2.1) и (2.2); $k=21$ 2. Таблица Трисемуса, ключевое слово – собственное имя
7	русский	1. Шифр Порты 2. Шифр Цезаря с ключевым словом, ключевое слово – собственная фамилия
8	английский	1. Шифр Цезаря с ключевым словом, ключевое слово – собственная фамилия, $a=24$ 2. Таблица Трисемуса, ключевое слово – собственное имя
9	немецкий	1. На основе соотношений (2.1) и (2.2); $k=7$ 2. Таблица Трисемуса, ключевое слово – enigma
10	польский	1. На основе соотношений (2.1) и (2.2); $k=28$ 2. Шифр Порты
11	белорусский	1. Шифр Цезаря с ключевым словом, ключевое слово – инфарматыка, $a=2$ 2. Таблица Трисемуса, ключевое слово – собственное имя

12	русский	1. Шифр Цезаря с ключевым словом, ключевое слово – безопасность 2. Таблица Трисемуса, ключевое слово – безопасность
13	английский	1. На основе аффинной системы подстановок Цезаря; $a=6$ , $b=7$ 2. Таблица Трисемуса, ключевое слово – security
14	немецкий	1. Виженера, ключевое слово – собственная фамилия 2. Шифр Порты
15	польский	1. Виженера, ключевое слово – bezpieczeństwo 2. На основе соотношений (2.1) и (2.2); $k=20$

- формировать гистограммы частот появления символов для исходного и зашифрованного сообщений;
- оценивать время выполнения операций зашифрования/расшифрования (напоминание: во многих языках программирования есть встроенные методы для замеров времени; при отсутствии такового в используемом языке можно воспользоваться разностью двух дат (например, в миллисекундах: время после выполнения программы – время до начала выполнения преобразования)).

При анализе полученных гистограмм можно сопоставить полученные данные с аналогичными результатами выполнения лабораторной работы №2 из [1].

Если указанный в таблице язык исходного текста не известен разработчику программного средства, можно взять документ на требуемом языке и воспользоваться доступным электронным переводчиком (возникающие при этом отдельные семантические неточности не следует считать существенным недостатком выполняемого анализа).

2. Результаты оформить в виде отчета по установленным правилам.

## ВОПРОСЫ ДЛЯ КОНТРОЛЯ И САМОКОНТРОЛЯ

1. В чем заключается основная идея криптографических преобразований на основе шифров замены?
2. Привести классификационные признаки и дать сравнительную характеристику разновидностям подстановочных шифров.
3. Сколько разновидностей шифров, подобных шифру Цезаря, можно составить для алфавитов русского и белорусского языков?

4. Найти ключ шифра, с помощью которого получен шифртекст: *'byajhvfwbjyyfzgjcktljdfntkmyjcnm'*.

5. Расшифровать (с демонстрацией каждого шага алгоритма) текст  $C_i = 'qrscqcocqclc'$ , зашифрованный аффинным шифром Цезаря при  $N=26$ ,  $a=3$ ,  $b=5$ .

6. Зашифровать и расшифровать свою фамилию (на основе кириллицы), используя аффинный шифр Цезаря.

7. Можно ли использовать в качестве ключевого в шифре Виженера слово, равное по длине открытому тексту? Обосновать ответ.

8. По какому признаку можно определить, что текст зашифрован шифром Плейфера?

9. Имеются ли предпочтения в выборе размеров таблицы Трисемуса для виртуального алфавита мощностью 40:  $4 \times 10$ ?  $10 \times 4$ ?  $5 \times 8$ ?  $8 \times 5$ ?  $2 \times 20$ ?  $20 \times 2$ ?

10. Сравнить криптостойкость шифра Цезаря и шифра Виженера.

11. Охарактеризовать основные методы взлома подстановочных шифров.

## Лабораторная работа № 3

### Исследование криптографических шифров на основе перестановки символов

**Цель:** изучение и приобретение практических навыков разработки и использования приложений для реализации перестановочных шифров (работа рассчитана на 4 часа аудиторных занятий).

**Задачи:**

1. Закрепить теоретические знания по алгебраическому описанию, алгоритмам реализации операций зашифрования/расшифрования и оценке криптостойкости перестановочных шифров.
2. Ознакомиться с особенностями реализации и свойствами различных перестановочных шифров на основе готового программного средства (L\_LUX).
3. Разработать приложение для реализации указанных преподавателем методов перестановочного зашифрования/расшифрования.
4. Выполнить исследование криптостойкости шифров на основе статистических данных о частотах появления символов в исходном и зашифрованном сообщениях.
5. Оценить скорость зашифрования/расшифрования реализованных способов шифров.
6. Результаты выполнения лабораторной работы оформить в виде описания разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента.

### 3.1 ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Сущность перестановочного шифрования состоит в том, что, исходный текст ( $M$ ) и зашифрованный текст ( $C$ ) основаны на использовании одного и того же алфавита, а тайной или ключевой информацией является алгоритм перестановки.

Шифры перестановки относятся к классу *симметричных*. Элементами текста могут быть отдельные символы (самый распространённый случай), пары, тройки букв и так далее.

Классическими примерами перестановочных шифров являются *анаграммы*. Анаграмма (от греч. *ανα* – «снова» и *γραμμα* – «запись») – литературный приём, состоящий в перестановке букв (или звуков), что в результате дает другое слово или словосочетание, например: проездной–подрезной, листовка–вокалист, апельсин–спаниель.

В классической криптографии шифры перестановки делятся на два подкласса:

- шифры *простой* или *одинарной перестановки* – при зашифровании символы открытого текста  $M_i$  перемещаются с исходных позиций в новые (в шифртексте  $C_i$ ) один раз,
- шифры *сложной* или *множественной перестановки* – при зашифровании символы открытого текста  $M_i$  перемещаются с исходных позиций в новые (в шифртексте  $C_i$ ) несколько раз.

### **3.1.1. Шифры одинарной перестановки**

#### **3.1.1.1. Шифры простой перестановки**

Среди шифров рассматриваемого подкласса иногда выделяют *шифры простой перестановки* (или *перестановки без ключа*). Символы открытого текста  $M_i$  перемешиваются по каким-либо правилам. Формально каждое из таких правил может рассматриваться в качестве ключа.

Пример 1. Простейшим примером является запись открытого текста в обратной последовательности. Так, если  $M_i$  = «шифр перестановки», то  $C_i$  = «иквонатсереп рфиш». Если переставляются в соответствующем порядке пары букв, то  $C_i$  = «киованстрепе фрши». При более длинных сообщениях можно таким же образом перемещать целые слова или блоки слов.

Подобную перестановку можно трактовать как *транспозицию*.

В общем случае для использования шифров одинарной перестановки используется таблица, состоящая из двух строк: в первой строке записываются буквы, во второй – цифры  $J$ . Строки состоят из  $n$  столбцов. Буквы составляют шифруемое сообщение. Цифры  $J = j_1, j_2, \dots, j_n$ , где  $j_1$  – номер позиции в зашифрованном сообщении первого символа открытого текста, где  $j_2$  – номер позиции в зашифрованном сообщении второго символа открытого текста и т. д. Таким образом, порядок следования цифр определяется используемым правилом (ключом) перестановки символов открытого текста для получения шифрограммы.

Если предположить, что некоторое сообщение  $M_i$  состоит из букв от  $m_1$  до  $m_n$ , то рассматриваемую таблицу можно представить как показано ниже (таблица 3.1).

Таблица 3.1. Общий вид таблицы для шифра одинарной перестановки

$m_1$	$m_2$	...	$m_n$
-------	-------	-----	-------



$j_1$	$j_2$	...	$j_n$
-------	-------	-----	-------

В первую строку таблицы 3.1 могут записываться также числа в порядке возрастания от 1 до  $n$ . Понятно, что эти числа соответствуют позициям букв в открытом тексте.

Процедура расшифрования также основана на использовании таблиц перестановки. Эти таблицы строятся на основе таблиц вида 3.1.

Пример 2. Пусть  $M_i$  = «кибервойны», здесь  $n = 10$ . Далее принимаем правило (ключ) перестановки:  $j_1=5, j_2=3, j_3=1, j_4=6, j_5=4, j_6=2, j_7=10, j_8=7, j_9=8, j_{10}=9$ .

Составим таблицу для зашифрования сообщения в форме табл. 3.1.

Таблица 3.2

к	и	б	е	р	в	о	й	н	ы
5	3	1	6	4	2	10	7	8	9

Представим эту таблицу только числами.

Таблица 3.3.

1	2	3	4	5	6	7	8	9	10
5	3	1	6	4	2	10	7	8	9

В соответствии с принятым ключом зашифрованное сообщение будет иметь вид:  $C_i$  = «бвиркейныю».

Легко подсчитать, что при отсутствии повторяющихся букв в шифруемом сообщении длиной  $n$  символов всего существует  $n!$  неповторяющихся ключей.

Для расшифрования сообщения, следуя логике рассмотренных процедур зашифрования, нам нужно также составить таблицу, первой строкой которой будет зашифрованный текст (таблица 3.4.). Здесь применяется примерно такой же подход, как и в шифрах подстановки.

Таблица 3.4.

б	в	и	р	к	е	й	н	ы	о
1	2	3	4	5	6	7	8	9	10

Таблицу 3.4 дополним 3-ей строкой, числа в столбцах которой соответствуют первой строке таблицы 3.3, одновременно составляя неизменную пару: 1 соответствует 3, 2 – 6 и т.д. (см. табл. 3.5).

Таблица 3.5

б	в	и	р	к	е	й	н	ы	о
1	2	3	4	5	6	7	8	9	10
3	6	2	5	1	4	8	9	10	9

Теперь расшифрованному сообщению «бвиркейные» будет соответствовать обратная перестановка: символы первой строки таблицы 3.5 нужно расположить в порядке в соответствии с 3-й строкой: 1 – «к», 2 – «и» и т. д.

Для использования на практике рассмотренный метод зашифрования/расшифрования не очень удобен. При больших значениях  $n$  приходится работать с таблицами, состоящими из большого числа столбцов. Кроме того, для сообщений разной длины необходимо создавать разные таблицы перестановок.

Следует также отметить сходство рассмотренных алгоритмов зашифрования/расшифрования и алгоритмов перемежения, которые изучались и анализировались в лабораторной работе №7 из [1].

### 3.1.1.2. Шифры простой блочной перестановки

Указанные шифры строятся по тем же правилам, что и шифры простой перестановки. Блок должен состоять из 2-х или более символов. Если общее число таких символов в сообщении не кратно длине сообщения, то последний блок можно дополнить произвольными знаками.

Пример 3. Пусть  $M_i$  = «кибервойны», примем длину блока, равную 2.

Для зашифрования построим таблицу (табл. 3.6).

Таблица 3.6

ки	бе	рв	ой	ны
5	1	4	2	3

В соответствии с табл. 3.6 получим  $C_i$  = «беойнырвки». Расшифрование производится по правилам, схожим с правилами для шифров простой перестановки.

### 3.1.1.3. Шифры маршрутной перестановки

Основой современных шифров рассматриваемого типа является геометрическая фигура. Обычно прямоугольник или прямоугольная матрица. В ячейки этой фигуры по определенному маршруту (слева-направо, сверху-вниз или каким-либо иным образом) записывается открытый текст. Для получения

шифrogramмы нужно записать символы этого сообщения в иной последовательности, т.е. по иному маршруту (см. аналогию с методами перемежения/деперемежения данных в лабораторной работе №7 [1]).

**Шифр Скитала (Сцитала).** Известно, что в V веке до н. э. в Спарте существовала хорошо отработанная система секретной военной связи. Для этого использовался специальный жезл «скитала» (греч. σκυτάλη – первое, вероятно, простейшее криптографическое устройство, реализующее метод перестановки (рис. 3.1).



Рисунок 3.1 – Скитала [15]

Для зашифрования и расшифрования необходимо было иметь абсолютно одинаковые жезлы. На такой предмет наматывалась пергаментная лента. Далее на эту ленту построчно наносился текст. Для расшифрования ленту с передаваемым сообщением нужно было намотать так же, как и при нанесении открытого текста. Подобным образом работает шифр, который иллюстрирует пример на рисунке 3.5 в [2].

Следуя вышеприведенным рассуждениям, может отождествить скитала с таблицей размерами:  $k$  – количество столбцов,  $s$  – количество строк. Поскольку при регулярном обмене данными сообщения часто имеют разную длину, то оба этих параметра за неизменяющийся ключ взять неудобно. Поэтому обычно в качестве известного каждой стороне ключа выбирается один из них (часто это  $s$ ), а второй вычисляется на основе известного и длины  $n$  сообщения  $M_i$ :

$$k = [(n - 1)/s] + 1. \quad (3.1)$$

При этом слагаемое в квадратных скобках должно быть целым числом [15].

Нетрудно себе представить аналогию между Скитала и таблицей, которая «намотана» на цилиндр.

При использовании шифра Скитала для формирования шифртекста сначала выбирается 1-ая буква открытого текста, затем  $(k+1)$ -буква,  $(2k+1)$ -буква и т.д., для некоторого  $k$ , равного числу букв в каждой строке скиталы. Значение  $k$  является постоянной величиной для данной скиталы,

**Организация маршрутной перестановки.** Уже упоминавшаяся маршрутная перестановка (записываем сообщение по строкам, считываем – по столбцам матрицы) можно усложнить и считывать не по столбцам, а по спирали (рис. 3.2,а), зигзагом (рис. 3.2,б), змейкой (рис. 3.2,в) или каким-то другим способом (см. рис. 3.2). Такие способы шифрования несколько усложняют процесс, однако усиливают криптостойкость шифра.

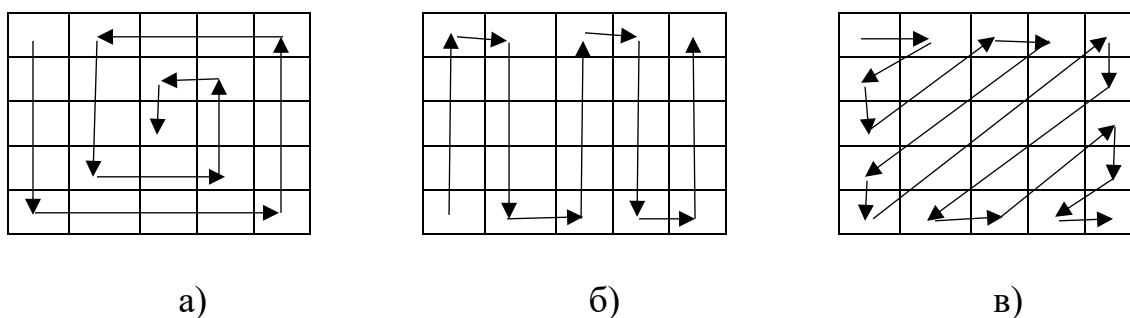


Рисунок 3.2 – Графическое представление методов маршрутной перестановки

Маршруты могут быть значительно более изощренными. Например, обход конем шахматной доски таким образом, чтобы в каждой клетке конь побывал один раз. Один из таких маршрутов был найден Л. Эйлером в 1759 г. Для примера на рис. 3.3 показан такой маршрут для обхода таблицы размером 5 x 4.

Не менее занимательным и не менее сложным является организация маршрутов на основе «магических квадратов» – квадратных матриц со вписанными в каждую клетку неповторяющимися последовательными числами от 1, сумма которых по каждому столбцу, каждой строке и каждой диагонали дает одно и то же число.

Создание новых оригинальных маршрутов приветствуется и поощряется при выполнении данной лабораторной работы.

### 3.1.1.4. Шифр вертикальной перестановки

Данный шифр является разновидностью шифра маршрутной перестановки. К особенностям вертикального шифра можно отнести следующие:

- количество столбцов в таблице фиксируется и определяется длиной ключа;
- маршрут вписывания: слева-направо, сверху-вниз;
- шифрограмма выписывается по столбцам в соответствии с их нумерацией (ключом).

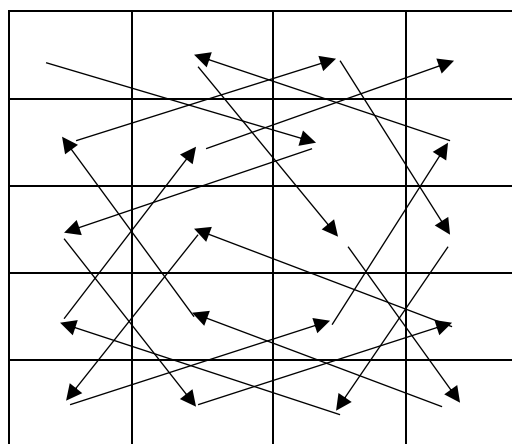


Рисунок 3.3 – Пример маршрута «обход конем»

Ключ может задаваться в виде текста (слова или словосочетания). Лексикографическое местоположение символов в ключевом выражении определяет порядок считывания столбцов.

Пример 4. Например, ключом является слово «крипто». Во-первых, это означает, что количество столбцов  $k$  в таблице должно быть равно длине ключа, т. е. – 6. Если вспомним порядок букв из ключевого слова в алфавите, то последовательность считывания столбцов будет следующим: 2, 5, 1, 4, 6, 3.

Необходимо зашифровать сообщение  $M_i$  = «шифр вертикальной перестановки»;  $n = 30$ .

Строим основную таблицу 5x6 (табл. 3.7), в которую по строкам будет записано исходное сообщение.

Считывая информацию из таблицы по столбцам в соответствии с ключом, получим шифрограмму  $C_i$  = «фтирошелпава тириоевирыен кйск».

Таблица 3.7

$\kappa$	$p$	$u$	$n$	$m$	$o$
2	5	1	4	6	3
ш	и	ф	р		в

е	р	т	и	к	а
л	ь	н	о	й	
п	е	р	е	с	т
а	н	о	в	к	и

### 3.1.2. Шифры множественной перестановки

Особенностью шифров данного подкласса является минимум двукратная перестановка символов шифруемого сообщения. В простейшем случае это может задаваться перемешиваем не только столбцов (как в примере 4), но и строк. Таким образом, этот случай соответствует использованию двух основных ключей: длина одного из них равна числу столбцов, другого – числу строк. К ключевой информации мы можем относить также способы вписывания сообщения и считывания отдельных символов из текущего столбца матрицы.

Пример 5. Предположим, что (в продолжение к последнему примеру) вторым ключом будет «слово» или 5, 2, 3, 1, 4 (одинаковым буквам «о» мы присвоили последовательные числа).

Предыдущая таблица несколько видоизменится и примет следующий вид (табл. 3.8).

Таблица 3.8

ключи		<i>к</i>	<i>р</i>	<i>и</i>	<i>п</i>	<i>т</i>	<i>о</i>
		2	5	1	4	6	3
<i>с</i>	5	ш	и	ф	р		в
<i>л</i>	2	е	р	т	и	к	а
<i>о</i>	3	л	ь	н	о	й	
<i>в</i>	1	п	е	р	е	с	т
<i>о</i>	4	а	н	о	в	к	и

Для удобства отсортируем последовательно строки в соответствии с ключом (табл. 3.9).

Таблица 3.9

ключи		<i>к</i>	<i>р</i>	<i>и</i>	<i>п</i>	<i>т</i>	<i>о</i>
		2	5	1	4	6	3

<i>в</i>	1	п	е	р	е	с	т
<i>л</i>	2	е	р	т	и	к	а
<i>о</i>	3	л	ь	н	о	й	
<i>о</i>	4	а	н	о	в	к	и
<i>с</i>	5	ш	и	ф	р		в

И столбцы – в соответствии с ключевым словом «слово».

Таблица 3.10

ключи		<i>и</i>	<i>к</i>	<i>о</i>	<i>п</i>	<i>р</i>	<i>т</i>
		1	2	3	4	5	6
<i>в</i>	1	е	т	р	е	с	п
<i>л</i>	2	и	а	т	р	к	е
<i>о</i>	3	о		н	ь	й	л
<i>о</i>	4	в	и	о	н	к	а
<i>с</i>	5	р	в	ф	и		ш

Получим итоговую шифрограмму  $C_i$  = «еиоврта ивртноферьнскийк пелаш».

Шифры гаммирования рассматриваются как самостоятельный класс. Такие шифры схожи с перестановочными тем, что в обоих случаях можно использовать табличное представление выполняемых операций на основе ключей. Вместе с тем, шифры гаммирования имеют много общего с подстановочными шифрами, поскольку на самом деле при зашифровании происходит подмена одних символов на другие.

Полезную информацию о классе рассмотренных шифров можно найти в [16, 17].

## 3.2 ПРАКТИЧЕСКОЕ ЗАДАНИЕ

*Рекомендация!* Перед выполнением практического задания целесообразно освежить практические навыки использования и особенностями функционирования программного средства *L\_LUX*, реализующего перестановочные (и другие) методы зашифрования/расшифрования текстовой информации и являющегося приложением на компакт-диске к [5].

Обратим внимание на использование «горячих» клавиш для реализации некоторых операций:

Ctrl + F3 – зашифрование на основе простой перестановки,

Shift + F3 – расшифрование на основе простой перестановки,  
 Shift + Ctrl + F1 – вывод гистограмм (частотных параметров символов) исходного и зашифрованного сообщений,  
 Shift + Ctrl + F2 – вывод гистограмм (частотных параметров символов) зашифрованного и расшифрованного сообщений.

### Основное задание.

1. Разработать авторское приложение в соответствии с целью лабораторной работы. Приложение должно реализовывать следующие операции:

- выполнять зашифрование/расшифрование текстовых документов (объемом не менее 500 знаков) созданных на основе алфавита языка в соответствии с нижеследующей таблицей вариантов задания; при этом следует использовать шифры подстановки из третьего столбца данной таблицы;

#### Варианты задания

Вариант	алфавит	шифр
1	белорусский	1. Маршрутная перестановка (маршрут: запись – по строкам, считывание – по столбцам таблицы; параметры таблицы – по указанию преподавателя) 2. Множественная перестановка, ключевые слова – собственные имя и фамилия
2	русский	1. Маршрутная перестановка (маршрут: по спирали; параметры таблицы – по указанию преподавателя) 2. Множественная перестановка, ключевые слова – собственные имя и фамилия
3	английский	1. Маршрутная перестановка (маршрут: зигзагом; параметры таблицы – по указанию преподавателя) 2. Множественная перестановка, ключевые слова – собственные имя и фамилия
4	немецкий	1. Маршрутная перестановка (маршрут: змейкой; параметры таблицы – по указанию преподавателя) 2. Множественная перестановка, ключевые слова – собственные имя и фамилия
5	польский	1. Маршрутная перестановка (маршрут: запись – по столбцам, считывание – по строкам таблицы; параметры таблицы – по указанию преподавателя) 2. Множественная перестановка, ключевые слова – собственные имя и фамилия



- формировать гистограммы частот появления символов для исходного и зашифрованного сообщений;
- оценивать время выполнения операций зашифрования/расшифрования (напоминание: во многих языках программирования есть встроенные методы для замеров времени; при отсутствии такового в используемом языке можно воспользоваться разностью двух дат (например, в миллисекундах: время после выполнения программы – время до начала выполнения преобразования).

Ниже представлен (Листинг 3.1) пример кода программы (класса *Encryption*) для зашифрования сообщения на основе табличного представления сообщений.

```
class Encryption{
{
    private int[] key = null;
    public void SetKey(string[] _key)
    {
        key=new int[_key.Length];
        for(int i=0;i<_key.Length;i++)
            key[i] = Convert.ToInt32(_key[i]);
    }
    public string Encrypt(string input)
    {
        for(int i=0;i<input.Length % key.Length;i++) input +=
input[i];

        string result = "";
        for(int i=0;i<input.Length;i+=key.Length)
        {
            char[] transposition = new char[key.Length];
            for(int j=0;j<key.Length;j++)
                transposition[key[j]-1]=input[i+j];
            for(int j=0;j<key.Length;j++)
                result += transposition[j];
        }
        return result;
    }
    public string Decrypt(string input)
    {
        string result = "";
        for(int i=0;i<input.Length;i+=key.Length)
        {
            char[] transposition = new char[key.Length];
            for(int j=0;j<key.Length;j++)
```

```

        transposition[j] = input[i + key[j] - 1];
        for(int j=0;j<key.Length;j++) result += transposi-
tion[j];
    }
    return result;
}
}

```

Листинг 3.1. Пример кода программы для зашифрования сообщения на основе табличного представления сообщений

При анализе полученных гистограмм можно сопоставить полученные данные с аналогичными результатами выполнения лабораторной работы №2 из [1] и лабораторной работы №2 настоящего пособия.

Если указанный в таблице язык исходного текста не известен разработчику программного средства, можно взять документ на требуемом языке и воспользоваться доступным электронным переводчиком (возникающие при этом отдельные семантические неточности не следует считать существенным недостатком выполняемого анализа).

2. Результаты оформить в виде отчета по установленным правилам.

## ВОПРОСЫ ДЛЯ КОНТРОЛЯ И САМОКОНТРОЛЯ

1. В чем заключается основная идея криптографических преобразований на основе шифров перестановки?
2. Привести классификационные признаки и дать сравнительную характеристику разновидностям перестановочных шифров.
3. Сколько разновидностей шифров, подобных шифру Цезаря, можно составить для алфавитов русского и белорусского языков?
4. Охарактеризовать криптостойкость перестановочных и подстановочных шифров.
5. Привести примеры дать характеристику перестановочным шифрам, не рассмотренным в материалах к данной лабораторной работе.
6. Имеются ли предпочтения в выборе размеров используемой таблицы для перестановочных шифров?
7. Охарактеризовать основные методы взлома перестановочных шифров.

## Лабораторная работа № 4

### Изучение устройства и функциональных особенностей шифровальной машины «Энигма»

**Цель:** изучение и приобретение практических навыков разработки и использования приложений для реализации перестановочных шифров (рассчитана на 4 часа аудиторных занятий).

**Задачи:**

1. Закрепить теоретические знания по алгебраическому описанию, алгоритмам реализации операций зашифрования/расшифрования и оценке криптостойкости подстановочно-перестановочных шифров.
2. Изучить структуру, принципы функционирования, реализацию процедур зашифрования сообщений в машинах семейства Энигма.
3. Изучить и приобрести практические навыки выполнения криптопреобразований информации на платформе Энигма, реализованной в виде симуляторов.
4. Получить практические навыки оценки криптостойкости подстановочных и перестановочных шифров на платформе Энигма.
5. Результаты выполнения лабораторной работы оформить в виде отчета проведенных исследований, методики выполнения практической части задания и оценки криптостойкости шифров.

#### 4.1 ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

##### *4.1.1 Краткая историческая информация*

Идея создания шифровального устройства высказана голландцем Гуго Кох де Дельфтью (в некоторых источниках – Гуго Александр Кох, Hugo Alexander Koch) еще в 1919 г. В 1920 году он же изобрел первую роторную шифровальную машинку. Параллельно с этим немец Артур Шербиус (Arthur Scherbius) изучал проблему криптостойкости (в нашем современном понимании) шифровальных машин. Он же получил патент на такую машину, которая получила название «Enigma» (от греч. – загадка). Основная особенность Энигмы – все знали в то время алгоритм шифрования, но никто не мог подобрать нужный ключ.

Первая шифровальная машина, *Enigma A*, появилась на рынке в 1923 году. Это была большая и тяжелая машина со встроенной пишущей машинкой и ве-

сом около 50 кг. Вскоре после этого была представлена *Enigma B*, очень похожая на *Enigma A*. Вес и размеры этих машин сделали их непривлекательными для использования в военных целях.

По достоинству шифровальную машину оценили в немецкой армии. В 1925 году её принял на вооружение сначала военно-морской флот (модель *Funkschlussen C*), а в 1930-м – и Вермахт (*Enigma I*). Общее количество шифраторов, произведённых до и во время Второй мировой войны, превысило 100 тысяч. Применялись они всеми видами вооружённых сил Германии, а также военной разведкой и службой безопасности.

Идея коллеги А. Шербиуса, Вилли Корна (Willi Korn), позволила создать компактную и намного более легкую *Enigma C*. Особенностью этой модели было наличие ламповой панели. В 1927 году *Enigma D* была представлена и коммерциализирована в нескольких версиях с различными роторами и продана военным и дипломатическим службам многих стран Европы. В *Enigma D* было три обычных ротора и один отражатель (рефлектор), которые можно было установить в одном из 26 положений (по числу букв используемого алфавита). Именно эта модель стала основным прототипом многих известных версий машин Энигма, которые использовала Германия в годы Второй Мировой войны.

#### **4.1.2 Конструкция и принцип функционирования Энигмы**

Машина Энигма – это электромеханическое устройство. Как и другие роторные машины, Энигма состоит из комбинации механических и электрических подсистем.

Механическая часть включает в себя клавиатуру, набор вращающихся дисков – роторов, – которые расположены вдоль вала и прилегают к нему, и ступенчатого механизма,двигающего один или несколько роторов при каждом нажатии на клавишу. Электрическая часть, в свою очередь, состояла из электрической схемы, соединяющей между собой клавиатуру, коммутационную панель, лампочки и роторы (для соединения роторов использовались скользящие контакты).

На рис. 4.1 показана фотография одной из моделей Энигмы с указанием месторасположения основных модулей машины. Как видно на этом рисунке, Энигма состоит из 5 основных блоков:

- панели механических клавиш, 1 (дают сигнал поворота роторных дисков);
- трех (или более) роторных дисков, 2, каждый имеет контакты по сторонам, по 26 на каждую, которые коммутируют в случайном порядке; по окружности нанесены буквы латинского алфавита либо числа;

- рефлектора, 3 (имеет контакты с крайним слева ротором);
- коммутационной панели, 4 (служит для того, чтобы дополнительно менять местами электрические соединения (контакты) двух букв);
- панели в виде электрических лампочек, 5; индикационная панель с лампочками служит индикатором выходной буквы в процессе шифрования.



Рисунок 4.1 Одно из моделей (трехроторная) Энигмы [18]

Конкретный механизм мог быть разным, но общий принцип был таков: при каждом нажатии на клавишу самый правый ротор сдвигается на одну позицию, а при определённых условиях сдвигаются и другие роторы. Движение роторов приводит к различным криптографическим преобразованиям при каждом следующем нажатии на клавишу на клавиатуре, т.е. зашифрование/расшифрование сообщений основано на выполнении ряда замен (подстановок) одного символа другим. Идея А. Шербиуса состояла в том, чтобы добиться этих подстановок электрическими связями.

Механические части двигались и замыкая контакты, образовывали меняющийся электрический контур. При нажатии на клавишу клавиатуры контур замыкается, ток проходит через созданную (для зашифрования/расшифрования одного конкретного символа сообщения) цепь и в результате включает одну из набора лампочек, отображающую искомую букву шифртекста (или расшифрованного сообщения). На рис. 4.2 показаны упрощенная конструкция ротора (а) и рефлектора (б). Замыкание цепи происходило за счет рефлектора.

На рис. 4.3 схематично показано, как некоторая буква (например, «а») будет зашифрована другой буквой (например, «g»), а следующая за ней буква сообщения (также «а») – уже буквой «с».

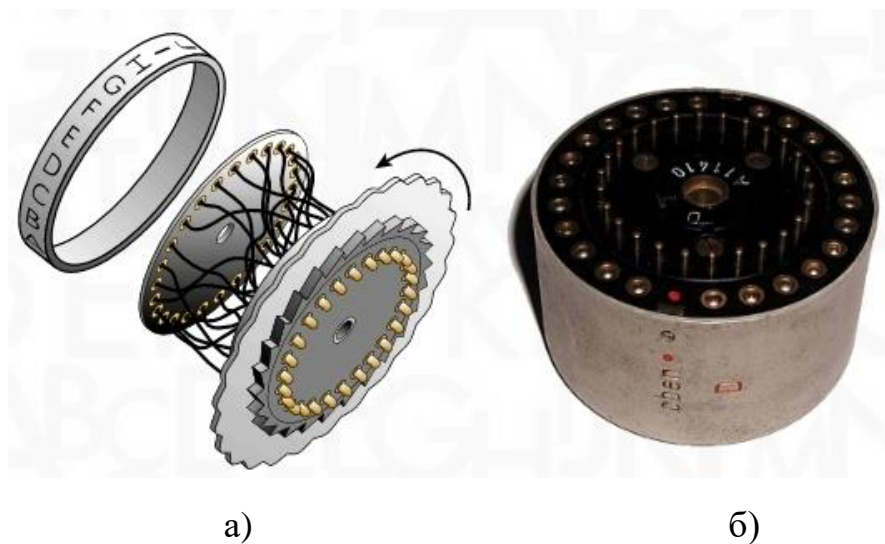


Рисунок 4.2 Упрощенная конструкция ротора (а) и рефлятора (б) Энигмы

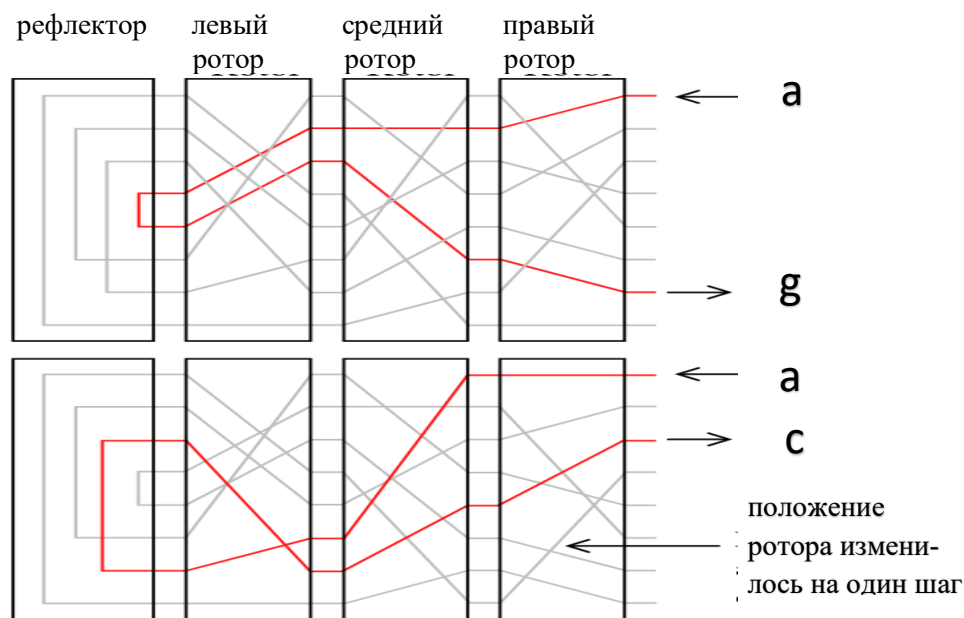


Рисунок 4.3 Пояснение к принципу шифрования путем формирования электрической цепи [19]

Отметим, что на рис. 4.3 электрическая цепь не представлена в виде замкнутой, поскольку не показаны части коммутационной панели и электрическая лампочка. Замкнутые электрические цепи хорошо иллюстрирует рис. 4.4.

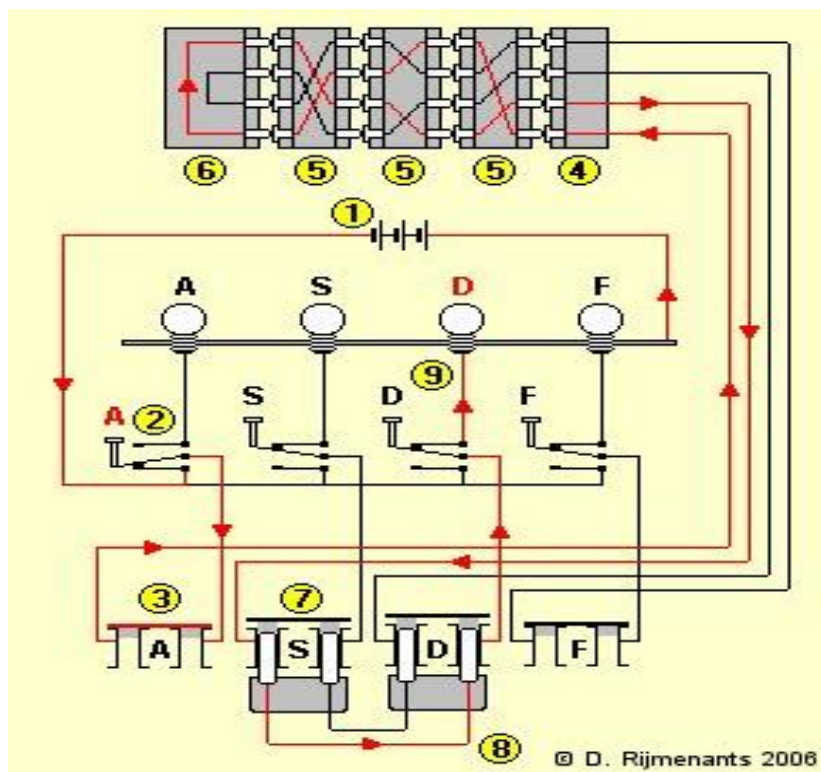


Рисунок 4.4 Пояснение к принципу формирования зашифрованного символа с помощью замкнутой электрической цепи [18]

Замкнутую цепь составляют: батарея 1 (это могут быть и иные источники питания), нажатая двунаправленная буквенная клавиша, 2, разъем коммутационной панели, 3 (как видим, в одном случае – буква «а» – коммутационного перехода на другую букву нет), входной разъем (входное колесо) 4 роторного модуля, роторный модуль 5 (состоит из трех роторов, как в версии Энигмы для Вермахта, *Wehrmacht Enigma M3*, или четырех – в версии Энигмы для военно-морского флота, *Kriegsmarine Enigma M4*), рефlector 6. Последний возвращает ток (цепь) по другому пути через узлы те же узлы, «зажигая» на ламповой панели букву «D», к другому полюсу батареи. Обратим внимание, что обратная часть цепи уже проходит с учетом выполненной коммутации (7 и 8).

Отметим также, что клавиатура соответствовала немецкой раскладке *QWERTZ*.

### 4.1.3. Шифры Энигмы

Во время Второй мировой войны немецкие операторы использовали специальную (тайную) шифровальную книгу для установки роторов и настроек колец. Для каждого сообщения они выбирали случайную стартовую позицию, например, *WZA* и случайный ключ сообщения, например, *SXT*. Затем оператор устанавливал роторы в стартовую позицию *WZA* и зашифровывал ключ сообщения *SXT*. Предположим, что в результате зашифрования ключа получится *UHL*. Далее оператор ставил ключ сообщения (*SXT*) как начальную позицию роторов и зашифровывал собственно сообщение. После этого он отправлял стартовую позицию (*WZA*) и зашифрованный ключ (*UHL*) вместе с сообщением. Получатель устанавливал стартовую позицию в соответствии с первой трехграммой (*WZA*) и, расшифровывая вторую триграмму (*UHL*), извлекал исходный ключ (*SXT*). Далее получатель использовал этот ключ как стартовую позицию для расшифрования шифртекста. Обычно срок действия ключей составлял один день.

Военная модель Энигмы использовала только 26 букв. Прочие символы заменялись редкими комбинациями букв. Пробел пропускаться либо заменялся на «X». Символ «X» также использовался для обозначения точки либо конца сообщения. Некоторые особые символы использовались в отдельных вооруженных частях, например, *Wehrmacht* заменял запятую двумя символами *ZZ* и вопросительный знак — словом *FRAGE* либо буквосочетанием *FRAQ*, а *Kriegsmarine M4* запятой соответствовала буква «Y».

Как мы отмечали выше, Энигма строится на основе подстановочных шифров, подобных на шифр Цезаря, в котором, как известно, ключ сообщения, который должен знать получатель, — это просто смещение между двумя алфавитами. Принято считать, что в основе шифра Энигмы лежит *динамический шифр Цезаря*.

Более сложная система использует случайный ряд символов для нижнего алфавита. Принцип, положенный в основу этой «случайности», имеет много общего с перестановочными шифрами. Например, ниже показан принцип подстановки, основанный на взаимной перестановке во втором (нижнем) алфавите в 13 парах символов, расположенных случайным образом:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
J	W	M	R	Z	L	N	T	U	A	O	F	C	G	K	S	Y	D	P	H	I	X	B	V	Q	E

Этот принцип случайности использовался и при изготовлении роторов и рефлекторов для Энигмы. Всего за время Второй мировой войны немцами было изготовлено восемь роторов и четыре рефлектора, но одновременно могло использоваться ровно столько, на сколько была рассчитана машина.



Техническую спецификацию на все произведенные роторы и рефлекторы можно найти в [12, 21]. Ниже на рис. 4.5 и 4.6 представлены эти спецификации соответственно на роторы и на рефлекторы.

Достаточно подробная информация об основных особенностях и обстоятельствах патентования, разработки и сферах использования практически всех (или большинства) версий Энигмы содержится в [22].

INPUT	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Rotor I	E	K	M	F	L	G	D	Q	V	Z	N	T	O	W	Y	H	X	U	S	P	A	I	B	R	C	J
Rotor II	A	J	D	K	S	I	R	U	X	B	L	H	W	T	M	C	Q	G	Z	N	P	Y	F	V	O	E
Rotor III	B	D	F	H	J	L	C	P	R	T	X	V	Z	N	Y	E	I	W	G	A	K	M	U	S	Q	O
Rotor IV	E	S	O	V	P	Z	J	A	Y	Q	U	I	R	H	X	L	N	F	T	G	K	D	C	M	W	B
Rotor V	V	Z	B	R	G	I	T	Y	U	P	S	D	N	H	L	X	A	W	M	J	Q	O	F	E	C	K
Rotor VI	J	P	G	V	O	U	M	F	Y	Q	B	E	N	H	Z	R	D	K	A	S	X	L	I	C	T	W
Rotor VII	N	Z	J	H	G	R	C	X	M	Y	S	W	B	O	U	F	A	I	V	L	P	E	K	Q	D	T
Rotor VIII	F	K	Q	H	T	L	X	O	C	B	J	S	P	D	Z	R	A	M	E	W	N	I	U	Y	G	V
Beta rotor	L	E	Y	J	V	C	N	I	X	W	P	B	Q	M	D	R	T	A	K	Z	G	F	U	H	O	S
Gamma rotor	F	S	O	K	A	N	U	E	R	H	M	B	T	I	Y	C	W	L	Q	P	Z	X	V	G	J	D

Рисунок 4.5 Спецификация на роторы Энигмы

reflector B	(AY) (BR) (CU) (DH) (EQ) (FS) (GL) (IP) (JX) (KN) (MO) (TZ) (VW)
reflector C	(AF) (BV) (CP) (DJ) (EI) (GO) (HY) (KR) (LZ) (MX) (NW) (TQ) (SU)
reflector B Dünn	(AE) (BN) (CK) (DQ) (FU) (GY) (HW) (IJ) (LO) (MP) (RX) (SZ) (TV)
reflector C Dünn	(AR) (BD) (CO) (EJ) (FN) (GT) (HK) (IV) (LM) (PW) (QZ) (SX) (UY)

Рисунок 4.6 Спецификация на рефлекторы Энигмы

Рассмотрим пример использования приведенных спецификаций.

Пример 4.1. В этом примере мы процедуру только зашифрования одной буквы («G»).

Предположим, что Энигма оснащена роторами I, II, III (см. рис. 4.5). Таким образом, правым ротором (R) является III приведенной спецификации. Предположим также, что каждый ротор находится в своем положении A, когда выполняется шифрование. Если взять информацию с этой страницы, указав фактическую разводку ротора, это означает, что правый ротор R производит подстановку в соответствии с переставленными буквами исходного алфавита, т. е. буква «G» будет заменена буквой «C»:

**ABCDEF G HIJKLMNOPQRSTUVWXYZ  
BDFHJL C PRTXVZNYEIWGAKMUSQO** ,

центральный ротор (M) или II заменяет букву «C» на букву «D»:

**AB C DEFGHIJKLMNOPQRSTUVWXYZ  
AJ D KSIRUXBLHWTMCQGZNPYFVOE** ,

левый ротор (L) или III – соответственно букву «D» на букву «F»:

**ABC D EFGHIJKLMNOPQRSTUVWXYZ  
EKM F LGDQVZNTOWYHXUSPAIBRCJ** .

Предположим далее, что используется рефлексор В (спецификация на рис. 4.6 – первая строка):

**ABCDE F GHIJKLMNOPQRSTUVWXYZ  
YRUHQ S LDPXNGOKMIEBFZCWVJAT** .

Обратим внимание на то, что рефлексор имеет только 13 соединений, т. е. имеется 13 пар подстановок: А - Y, В - R и т. д. В нашем примере произошла подстановка F -> S.

Ток теперь проходит обратный путь через три ротора в последовательности L-M-R.

Эффект преобразования левого ротора (обратный):

**ABCDEFGHIJKLMN OPQR S TUVWXYZ  
UWYGADFPVZBECKMTHX S LRINQOJ** ,

соответственно – среднего ротора (обратный):

**ABCDEFGHIJKLMN O P Q R S T U V W X Y Z**  
**A J P C Z W R L F B D K O T Y U Q G E N H X M I V S**

и, наконец, – правого ротора (обратный):

**A B C D E F G H I J K L M N O P Q R S T U V W X Y Z**  
**T A G B P C S D Q E U F V N Z H Y I X J W L R K O M**

После всех (в данном случае – 7) подстановок буква «G» будет зашифрована буквой «P».

Процедура расшифрования шифртекстов предусматривала настройку отражателя, роторов и коммутационной панели машины в соответствии с таблицами (книгами) и использованными при зашифровании паролями. Достаточно подробно информацию об этом можно получить в [23], комментариев к функционалу Энигмы и ее симулятору [24]. Подробное описание кода симулятора на языке Python содержится в книге [25].

### ***3.1.4 Оценка криптостойкости Энигмы***

Для получения общего представления об особенностях работы криптоаналитиков над шифрами Энигмы полезно ознакомиться с содержанием материалов в [26].

Как мы неоднократно подчеркивали, преобразование «Энигмы» для каждой буквы может быть определено математически как результат подстановок. Рассмотрим трехроторную модель Энигмы. Положим, что символом  $B$  обозначаются операции с использованием коммутационной панели, соответственно символы  $Re$  – отражателя, а  $L$ ,  $M$  и  $R$  – обозначают действия левых, средних и правых роторов соответственно. Тогда процесс зашифрования символа  $m$  с использованием некоторой ключевой информации  $K$  формально можно записать в следующем виде:

$$E_K = f(m, B, Re, L, M, R). \quad (4.1)$$

Чтобы оценить криптостойкость шифра, нужно учитывать все возможные настройки машины. Для этого необходимо рассмотреть следующие свойства Энигмы:

- выбор и порядок роторов,
- разводку (коммутацию) роторов,

- настройку колец на каждом из роторов,
- начальное положение роторов в начале сообщения,
- отражатель,
- настройки коммутационной панели.

Используются различные варианты подсчета всех возможных состояний перечисленных конструктивных модулей машины [27]. К сожалению для немцев, взломщики шифра союзников знали машину, роторы и внутреннюю разводку этих роторов. Поэтому им нужно было учитывать только возможные способы настройки Энигмы. Такая априорная информация о конструктивных особенностях устройства для шифрования (вспомним об основных постулатах О. Керкгоффса [2]) в нашем случае снижает уровень (теоретический) криптостойкости (до практического). Немецкие криптологи полагали, что один ротор может быть подключен  $4 \times 10^{26}$  различными способами. Сочетание трех роторов и отражателя позволяет получить астрономические цифры возможных вариантов подстановок. Для союзников, которые знали конструкции роторов, число различных вариантов существенно уменьшалось.

Рассмотрим пример для трехроторной Энигмы Вермахта с отражателем (по умолчанию – В, см. рис. 4.6) и выбором из 5 роторов. Использовались 10 штекерных кабелей на коммутационной панели (количество кабелей по умолчанию, поставляемых с машиной).

Чтобы выбрать 3 ротора из возможных 5, существует 60 комбинаций ( $5 \times 4 \times 3$ ). Каждый ротор (его внутренняя проводка) может быть установлен в любом из 26 положений. Следовательно, с 3 роторами имеется 17 576 различных положений ротора ( $26 \times 26 \times 26$ ). Кольцо на каждом роторе содержит маркировку ротора (что здесь неважно) и выемку, которая влияет на шаг перемещения расположенного левее ротора. Каждое кольцо может быть установлено в любом из 26 положений. Поскольку слева от третьего (наиболее левого) ротора нет ротора, на расчет влияют только кольца самого правого и среднего ротора. Это дает 676 комбинаций колец ( $26 \times 26$ ).

Коммутационная панель обеспечивает самый большой набор возможных настроек. Для первого кабеля одна сторона может иметь любое из 26 положений, а другая сторона – любое из 25 оставшихся положений (одна буква коммутируется с другой). Однако, поскольку комбинация и ее обратная сторона идентичны (АВ такая же, как ВА), мы должны игнорировать все двойные числа во всех возможных комбинациях для одного кабеля, предоставляя  $(26 \times 25) / (1! \times 2^1)$  или 325 уникальных способов коммутаций одним кабелем. Для двух кабелей: есть  $(26 \times 25)$  комбинаций – для первого кабеля и, поскольку два разъема уже используются, то получается  $(24 \times 23)$  комбинаций – для второго кабеля. Следуя этой простой логике, получается  $(26 \times 25 \times 24 \times 23) / (2! \times 2^2) = 44\,850$  уникальных способов коммутаций с использованием двух кабелей.

Для трех кабелей –  $(26 \times 25 \times 24 \times 23 \times 22 \times 21) / (3! \times 2^3) = 3\,453\,450$  комбинаций и так далее. Таким образом, с использованием 10 кабелей на коммутационной панели получаются  $150\,738\,274\,937\,250$  различных комбинаций. Формула, где  $n$  равно количеству кабелей, равна  $26! / (26 - 2n)! \cdot n! \cdot 2^n$ . Численно это дает:  $60 \times 17\,576 \times 676 \times 150\,738\,274\,937\,250 = 107\,458\,687\,327\,250\,619\,360\,000$  или  $1,07 \times 10^{23}$ .

Таким образом, практически рассматриваемая версия Энигмы (три ротора с выбором из 5 роторов, отражатель В и 10 штекерных кабелей для коммутационной панели) может быть настроена на  $1,07 \times 10^{23}$  различных состояний, что сопоставимо с 77-битным криптографическим ключом.

Добавление четвертого ротора (например, для *Naval Enigma M4*) для повышения его криптостойкости было практически бесполезным: неподвижный четвертый ротор «усложнил машину» только в 26 раз и вместе с тонким отражателем мог рассматриваться как настраиваемый отражатель с 26 положениями. Внедрение общего числа роторов в 8 единиц (на *Kriegsmarine M3*), а затем – на четырехроторной версии (*M4*) было гораздо более эффективным шагом. Они увеличили комбинации роторов с 60 до 336.

Оценим далее практический размер криптографического ключа (или его эквивалент) для четырехроторной версии *Kriegsmarine Enigma M4*. Эта машина использует 3 обычных ротора, выбранных из набора из 8. Это, как мы уже отметили, дает 336 комбинаций подключений роторов ( $8 \times 7 \times 6$ ). *M4* также имела специальный четвертый ротор, называемый *Beta* или *Gamma* (без кольца), который дает 2 варианта. Они не совместимы с другими роторами и подходят только как четвертый (самый левый) ротор. Четыре ротора могут быть установлены в любом из  $456\,976$  положений ( $26 \times 26 \times 26$ ). Рефлектор не меняется. Четвертый ротор был неподвижным. Версия *M4* была снабжена также 10 кабелями для коммутационной панели.

В сумме это дает:  $336 \times 2 \times 456\,976 \times 676 \times 150\,738\,274\,937\,250 = 31\,291\,969\,749\,695\,380\,357\,632\,000$  или  $3,1 \times 10^{25}$ , что сопоставимо с 84-битным ключом.

Проблема криптоанализа шифров Энигмы была экстраординарной (с учетом электромеханических конструкций устройств для криптоанализа, применяемых в то время). Исчерпывающий поиск всех возможных  $1,07 \times 10^{23}$  настроек (атака *brute force*) был невозможен в 1940-х годах, а его сопоставимый 77-битный ключ огромен даже для современных электронных систем. Чтобы дать представление о размере этого числа, представим, что у нас есть  $1,07 \times 10^{23}$  листов бумаги толщиной около 1 мм. Из этих листов можно сложить примерно  $70\,000\,000$  стопок бумаги, каждая из которых простирается от Земли до Солнца. Кроме того,  $1,07 \times 10^{23}$  дюйма равно  $288\,500$  световых лет.

## 4.2 ПРАКТИЧЕСКОЕ ЗАДАНИЕ

1. Ознакомиться с функционалом хотя бы одного (по согласованию с преподавателем) симулятора Энигмы:

1.1 Симулятор Энигмы M3 (M3 Enigma Simulator)

<https://cryptocellar.org/simula/m3/index.html>

1.2 Симулятор Энигмы M4 (M4 Enigma Simulator)

<https://cryptocellar.org/simula/m4/index.html>

1.3 Симулятор Энигмы Army/Air Force and the Railway

<https://cryptocellar.org/simula/enigma/index.html>

1.5 Симулятор Энигмы для Абвера (Abwehr Enigma Simulator)

<https://cryptocellar.org/simula/abwehr/index.html>

1.5 Симулятор Энигмы для Тирпица (T (Tirpitz) Enigma Simulator)

<https://cryptocellar.org/simula/tirpitz/index.html>

Произвести зашифрование сообщения (собственные имя, отчество, фамилия) при 8-10 различных настройках машины-симулятора. Оценить частотные свойства символов в шифртекстах и сравнить этот параметр с частотными свойствами символов для исходного текста.

2. Разработать приложение-симулятор шифровальной машины, состоящей из клавиатуры, трех роторов и отражателя. Типы роторов ( $L$ - $M$ - $R$ ) и отражателя  $Re$  следует выбрать из таблиц на рис. 4.5 и 4.6 в соответствии со своим вариантом, представленным в таблице 4.1. Крайний правый столбец этой таблицы показывает, на какое число шагов (букв,  $i$ ) перемещается соответствующий ротор при зашифровании одного (текущего) символа; число 0 означает перемещение соответствующего ротора на один шаг при условии, что расположенный правее ротор совершит один оборот.

Таблица 4.1

Вариант задания	$L$	$M$	$R$	$Re$	$L_i$ - $M_i$ - $R_i$
1	I	II	III	B	0-2-2
2	II	III	V	C	1-2-2
3	III	VII	I	B Dunn	1-0-1
4	IV	III	II	C Dunn	0-0-4
5	I	Beta	Gamma	B	3-1-3
6	II	Gamma	IV	C	1-1-1
7	Beta	Gamma	V	B Dunn	0-2-2
8	V	VI	VII	C Dunn	1-2-2
9	VIII	II	IV	B	1-0-1
10	Gamma	III	Beta	C	0-0-4

11	Beta	VIII	I	B Dunn	3-1-3
12	III	Gamma	V	C Dunn	1-1-2
13	VI	IV	II	B	1-2-2
14	II	Beta	VIII	C	0-2-2
15	VII	Gamma	II	B Dunn	1-2-2

С помощью разработанного приложения зашифровать сообщение в соответствии с п.1 практического задания, применив не менее 5 вариантов начальных установок роторов.

Оценить криптостойкость вашего варианта машины.

3. Результаты оформить в виде отчета по установленным правилам.

### **ВОПРОСЫ ДЛЯ КОНТРОЛЯ И САМОКОНТРОЛЯ**

1. Дать пояснение к структуре шифровальных машин Энигма.
2. На основе каких шифров строится машина Энигма?
3. Дать пояснение к принципам зашифрования сообщений.
4. Дать характеристику криптостойкости шифровальной машины Энигма.
5. Дать характеристику (с численными оценками) криптостойкости машины-симулятора на основе разработанного приложения.
6. Пояснить основные принципы расшифрования сообщений Энигмы.
7. Ваши предложения по модификации известных аналогов Энигмы?

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Урбанович, П. П. Лабораторный практикум по дисциплинам "Защита информации и надежность информационных систем" и "Криптографические методы защиты информации". В 2 ч. Ч. 1. Кодирование информации: учебно-методическое пособие для студентов учреждений высшего образования / П. П. Урбанович, Д. В. Шиман, Н. П. Шутько. – Минск: БГТУ, 2019. – 116 с.  
(Ресурс удаленного доступа: <https://elib.belstu.by/handle/123456789/29372>)
2. Урбанович, П. П. Защита информации методами криптографии, стеганографии и обфускации : учеб.-метод. пособие для студ. спец. 1-98 01 03 «Программное обеспечение информационной безопасности мобильных систем», направления спец. 1-40 05 01-03 «Информационные системы и технологии (издательско-полиграфический комплекс)», спец. 1-40 01 01 «Программное обеспечение информационных технологий» спец. 1-40 01 01 10 «Программирование Интернет-приложений». - Минск: БГТУ, 2016. - 220 с. (Ресурс удаленного доступа: <https://elib.belstu.by/handle/123456789/23763>)
3. Как определить простое число, Ресурс удаленного доступа: <https://www.kakprosto.ru/kak-66290-kak-opredelit-prostoe-chislo#ixzz60MRJ4m5m>
4. Шнайер, Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си/ Б. Шнайер. – М.: Издательство Триумф, 2003. – 816 с.
5. Бабаш, А. В. Информационная безопасность. Лабораторный практикум: уч. пособие/ А. В. Бабаш, Е. К. Баранова, Ю. Н. Мельникова. – 2-е изд., стер. – М.: КНОРУС, 2013. – 136 с.
6. Жуан Гомес. Мир математики. Т.2: Математики, шпионы и хакеры. Кодирование и криптография. – М.: Де Агостини, 2014. – 144 с.  
(Ресурс удаленного доступа: <https://www.litmir.me/br/?b=247091&p=1>)
7. Национальный корпус русского языка, Ресурс удаленного доступа: <http://www.ruscorpora.ru/new/corpora-intro.html>
8. Урбанович, П. П. Защита информации: конспект-лекция. Ч. 7 = Information Protection, Part 7: Basic cryptographic algorithms and standards / П. П. Урбанович. - Минск: БГТУ, 2019. – 58 с. – на англ. яз.  
(Ресурс удаленного доступа: <https://elib.belstu.by/handle/123456789/29342>)
9. Риксон, Фред Б. Коды, шифры, сигналы и тайная передача информации / Фред Б. Риксон - М.: АСТ: Астрель, 2011. – 656 с.
10. Leone Baptista Alberti. De componendis cyfris, Ресурс удаленного доступа: <http://www.apprendre-en-ligne.net/crypto/alberti/decifris.pdf>
11. Шифры замены, Ресурс удаленного доступа: <https://www.sites.google.com/site/anisimovkhv/learning/kripto/lecture/tema4#tabl41>



12. Бабаш, А. В. Криптография. Под. ред. В. П. Шерстюка, Э. А. Применко / А. В. Бабаш, Г. П. Шанкин. – М.: СОЛОН-ПРЕСС, 2007. – 512 с.
13. Кан, Д. Взломщики кодов / Д. Кан – М.: Центрполиграф, 2000. – 594 с.
14. Н. А. Гатченко, А. С. Исаев, А. Д. Яковлев – Криптографическая защита информации, уч. пособие, Санкт-Петербург – 2012.
15. Скитала, Ресурс удаленного доступа: <https://ru.wikipedia.org/wiki/Скитала>
16. Шифры перестановки, Ресурс удаленного доступа:  
<https://www.sites.google.com/site/anisimovkhv/learning/kripto/lecture/tema5>
17. Перестановочные шифры, Ресурс удаленного доступа:  
<http://nozdr.ru/games/quest/crypt/cipher/perestанovka>
18. Шифровальная машина Энигма, Ресурс удаленного доступа:  
<http://www.adsl.kirov.ru/projects/articles/2018/06/24/enigma/>
19. Алгоритм Энигмы, Ресурс удаленного доступа:  
<https://habr.com/ru/post/217331/>
20. Technical Details of the Enigma Machine, Ресурс удаленного доступа:  
<http://users.telenet.be/d.rijmenants/en/enigmatech.htm#rotors>
21. Technical Specification of the Enigma, Ресурс удаленного доступа:  
<http://www.codesandciphers.org.uk/enigma/rotorspec.htm>
22. Enigma Timeline. Under construction, Ресурс удаленного доступа:  
<https://www.cryptomuseum.com/crypto/enigma/timeline.htm>
23. Enigma Mashinen, Ресурс удаленного доступа:  
<https://yandex.by/video/prview/?filmId=4334530940425969248&noreask=1&path=wiard&text=симуляторы+Энигмы>
24. Jorge Luis Orejel. Enigma, Ресурс удаленного доступа:  
<https://www.codeproject.com/Articles/831015/ENIGMA>
25. Brian Neal. Py-Enigma Documentation. Release 0.1, 2017. – 29 p. (Ресурс удаленного доступа:  
<https://readthedocs.org/projects/py-enigma/downloads/pdf/latest/>)
26. Криптоанализ «Энигмы», Ресурс удаленного доступа:  
[https://ru.wikipedia.org/wiki/Криптоанализ\\_«Энигмы»](https://ru.wikipedia.org/wiki/Криптоанализ_«Энигмы»)
27. A. Ray Miller. The Cryptographic Mathematics of Enigma, Center for Cryptologic History National Security Agency, 2019  
(Ресурс удаленного доступа:  
[https://drive.google.com/file/d/1By1nea1BhLiNwCfykdmQAawkyh5QT\\_hr/view](https://drive.google.com/file/d/1By1nea1BhLiNwCfykdmQAawkyh5QT_hr/view))