1. Министерство образования и науки Российской Федерации
2. Санкт-Петербургский Политехнический Университет Петра Великого
3. —
4. Институт кибербезопасности и защиты информации

**КУРСОВАЯ РАБОТА**

1. «**Шифрование и дешифрование XOR**»
2. по дисциплине «Структуры данных»
3. Выполнил
4. студент гр. 4851001/20001 Смирнова В. А.
5. Козлов О. И.

<*подпись*>

1. Преподаватель
2. Доцент Семьянов П. В.

<*подпись*>

1. Санкт-Петербург
2. 2023

Оглавление

[1. Цель работы 3](#_Toc135172551)

[2. Постановка задачи 3](#_Toc135172552)

[3. Ход работы 4](#_Toc135172553)

[4. Выводы 49](#_Toc135172554)

[*Приложение* 51](#_Toc135172555)

## Цель работы

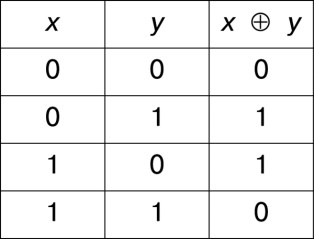
Изучить метод шифрования и дешифрования XOR с ключом, а также попытаться восстановить исходный текст после утери ключа.

## Задачи

* Написать программу, которая шифрует сообщение с ключом при помощи операции XOR;
* Написать программу, которая дешифрует сообщение без ключа при помощи операции XOR;
* Адаптировать программы под русский и английский языки, разные длины ключей и сообщений;
* Протестировать программу и сделать анализ конечных результатов;

*Теоретическая часть*

***XOR (или "исключающее ИЛИ")*** — это битовая операция, которая используется в криптографии для зашифрования данных. Она применяется к двум битам (цифрам в двоичной системе счисления) и возвращает результат, который будет равен 1 только в том случае, если только один из двух битов равен 1. Таблица истинности для XOR следующая:



*Рисунок 1 – операция XOR*

Таким образом при выполнении исключающего ИЛИ всегда будет нулевое значение, если переменные имели одинаковые значения. Особенность XOR в том, что одной и той же функцией можно как зашифровать данные, так и расшифровать их. Это простой метод шифрации данных, который может быть взломан достаточно быстро при наличии достаточно большого зашифрованного текста, или большого словаря паролей. Но тем не менее это уже можно применять для небольшой первоначальной защиты данных.

Для того, чтобы дешифровать текст, не зная ключа, необходимо найти его длину при помощи частотного анализа. Частотный анализ (также известный как подсчет букв) – это исследование [частоты букв](https://translated.turbopages.org/proxy_u/en-ru.ru.12b14139-645e7640-77c4868c-74722d776562/https/en.wikipedia.org/wiki/Letter_frequencies) или групп букв в [зашифрованном тексте](https://translated.turbopages.org/proxy_u/en-ru.ru.12b14139-645e7640-77c4868c-74722d776562/https/en.wikipedia.org/wiki/Ciphertext). Метод используется в качестве вспомогательного средства для взлома [классических шифров](https://translated.turbopages.org/proxy_u/en-ru.ru.12b14139-645e7640-77c4868c-74722d776562/https/en.wikipedia.org/wiki/Classical_cipher).

*Алгоритм частотного анализа для нахождения длины ключа шифрованного текста может быть следующим:*

1. Получите шифрованный текст, для которого вы хотите найти длину ключа.
2. Подготовьте таблицу для подсчета частотности символов или букв в тексте.
3. Разделите шифрованный текст на группы символов, где каждая группа содержит символы, расположенные на одинаковом смещении от начала текста, равном предполагаемой длине ключа. Например, если предполагаемая длина ключа равна 5, то первая группа будет содержать символы с позициями 1, 6, 11 и т.д., вторая группа - символы с позициями 2, 7, 12 и т.д.
4. Для каждой группы символов подсчитайте частотность каждого символа или буквы.
5. Оцените индекс совпадений для каждой группы символов, используя формулу:

*Индекс совпадений = Σ(fi \* (fi - 1)) / (n \* (n - 1))*

*где fi - частотность символа i в группе символов, n - общее количество символов в группе.*

1. Попробуйте различные предполагаемые длины ключа и выберите ту, для которой индекс совпадений наибольший. Обычно это будет длина ключа, при которой индекс совпадений будет близким к индексу совпадений для текстов на естественном языке (например, английском языке), который составляет около 0.057.
2. Повторите шаги с 3 по 6 для разных предполагаемых длин ключа и выберите длину ключа, при которой индекс совпадений наиболее близок к индексу совпадений для текстов на естественном языке.

Заметим, что для эффективного применения метода частотного анализа необходимо иметь достаточно длинный шифрованный текст.

## Ход работы

*Программа для XOR шифрования с ключом*

**

*Рисунок 2 – XOR шифратор*

В начале нами была написана программа, предназначенная для шифрования

сообщения с ключом при помощи операции XOR.

Ее принцип работы сводится к следующему:

1. Функция объявляет три указателя на файл: F.Start для входного файла, F.Coded для выходного файла и F.Key для файла с ключом.

2. Функция открывает входной файл в режиме двоичного чтения ("rb"), выходной файл в режиме двоичной записи ("wb") и файл с ключом в режиме двоичного чтения.

3. Она объявляет переменную структуры Sym, которая имеет два элемента: Sym.File и Sym.Key. Эти переменные будут содержать символы, считанные из входного файла и файла с ключом соответственно.

4. Функция определяет длину входного файла, перемещаясь в конец файла (SEEK\_END) с помощью fseek, а затем получает текущую позицию файла с помощью ftell.

5. Она выполняет ту же операцию для определения длины файла с ключом.

6. Функция входит в цикл, который продолжается до достижения конца входного файла (ftell(F.Start) != buf1).

7. Внутри цикла она считывает символ из входного файла с помощью fgetc и сохраняет его в Sym.File.

8. Если текущая позиция в файле с ключом (ftell(F.Key)) меньше длины файла с ключом (buf2), она считывает символ из файла с ключом и сохраняет его в Sym.Key.

9. Если текущая позиция в файле с ключом равна или превышает длину файла с ключом, это означает, что достигнут конец файла с ключом. В этом случае она перемещается в начало файла с ключом (fseek(F.Key, 0, SEEK\_SET)) и считывает символ из файла с ключом для повторного чтения с самого начала.

10. Функция выполняет операцию XOR между Sym.File и Sym.Key, чтобы зашифровать символ из входного файла.

11. Она записывает зашифрованный символ в выходной файл с помощью fputc.

12. Цикл продолжается до достижения конца входного файла.

13. Наконец, функция закрывает все открытые файлы с помощью fclose.

*Программа для XOR дешифрования без ключа*



Рисунок 3 – Программа для дешифрования зашифрованного сообщения без ключа XOR

После написанной программы для XOR шифрования сообщения, была создана программа, выполняющая дешифрование. Однако, задача была осложнена тем, что расшифровка должна проходить при помощи подбора ключа методом частотного анализа, описанного выше.

Принцип работы:

1. Сначала программа определяет длину ключа с помощью функции LenKey(). Эта функция анализирует зашифрованный текст и вычисляет индексы совпадения символов для разных длин ключей. Затем функция выбирает наиболее вероятную длину ключа, которая соответствует условию coincidence\_index\_avg[i] >= PROBABILITY (равно 0.057 для английских символов)

2. После определения длины ключа программа вызывает функцию FindKey, которая находит возможные символы ключа путем анализа частоты символов в зашифрованном тексте следующим образом:

В начале выполняется итерация по возможным позициям символов ключа: используется цикл for для перебора позиций символов в ключе от 0 до Length\_Key - 1. После этого создается массив block, который используется для анализа количества символов на каждой позиции.

Для каждой позиции символа в ключе:

* Выполняется выделение памяти для массива block размером (buf1 / Length\_Key + 1).
* Обнуляются элементы массива block с помощью memset.
* Инициализируется переменная index для отслеживания индекса в массиве block.

После этого необходимо провести анализ популярности для символов из массива block. Для этого используется цикл for для перебора всех символов на текущей позиции в зашифрованном тексте по следующему приницу:

* Читается символ из файла Coded.txt и записывается в соответствующий элемент массива block.
* Применяется операция XOR с пробелом (' ') для расшифровки символа.

Теперь выполняем подсчет популярности символов и формирование возможных ключей:

* Инициализируется переменная j для перебора всех возможных символов (от 0 до 255).
* Создаются массивы Kol и Alphabet размером 256 для подсчета количества символов и хранения символов алфавита соответственно.
* С помощью вложенных циклов while и for подсчитывается количество каждого символа в массиве block.
* Вызывается функция insertionSort для сортировки массивов Kol и Alphabet в порядке убывания количества символов.
* Выбираются первые GENERATION (константа) символов из массива Alphabet в порядке популярности и записываются в массив mbkey. Эти символы будут первой возможной комбинацией ключа, потому необходимо их сохранить в файл probably\_keys.txt.

Теперь проанализируем массив возможных символов ключа mbkey, устроенный следующим образом:



*Рисунок 4 – устройство массива mbkey*

Переберем возможные комбинации символов их этого массива:

* Первые по популярности символы в массиве mbkey (т.е. символы 1 4 7 в вышеуказанной схеме);
* Перестановки символов по длине ключа в функции print\_pattern\_chars

(Функция выводит комбинации символов ключа, где каждая колонка представляет символы на одной позиции ключа, и строки представляют различные комбинации символов);

* Очистка массива, от всех символов, кроме русских;
* Очистка массива, от всех символов, кроме английских;
* Только буквенные символы в массиве;

Причем, каждый символ комбинации проверяется на принадлежность к алфавиту или к особым символам. Если символ является сервисным, он заменяется пробелом.

3. В результате получим файл "probably\_keys.txt" который содержит все возможные ключи, из которых необходимо отобрать ключ, при котором зашифрованное сообщение будет дешифровано лучшим образом (будем определять при помощи счета букв после дешифровки сообщения. Чем больше пропорция количества букв в расшифрованном сообщении к длине зашифрованного сообщения, тем больше шанс того, что проверяем ключ искомый.

4. Получив наиболее вероятный ключ, расшифруем при помощи него операцией XOR зашифрованной сообщение и сохраним его в файл Generate\_TEXT.txt)

*Адаптация программы и анализ результатов*

Поскольку программа, используемая для дешифрования сообщения без ключа, основана на принципе частотного анализа, то для использования ее для русского и английского алфавита необходимо было провести следующую доработку:

1. Использования другого индекса совпадения для использования обоих языков. Для русских символов требуется индекс совпадения чуть ниже, чем для английских символов – 0.046. Это связано с тем, что русских символов больше, чем английских (33 против 26), по этой причине они распределены более равномерно.
2. Использование перебора возможных вариантов для более точного определения ключа, суть которого заключается в том, что мы будем перебирать возможные варианты из массива возможных символов ключа **mbkey.**

Возможны следующие варианты перебора ключа:

* Самые популярные символы в массиве mbkey;
* Перестановки символов по длине ключа;
* Очистка массива, от всех символов, кроме русских;
* Очистка массива, от всех символов, кроме английских;
* Массив, состоящий только из букв;

Данные комбинации ключей сохраняются в txt файл, который в дальнейшим можно будет использовать для дешифрования сообщения. В итоге найденным ключом считается тот, при помощи которого было дешифровано больше всего буквенных символов относительно всей длины зашифрованного сообщения.

*Тестирование программы и анализ результатов*

***Английские символы:***

**100 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | a | 2 | aa |
| 2 | up | 2 | up |
| 3 | joy | 3 | joy |
| 4 | frog | 2 | fg |
| 5 | tiger | 5 | !',: |
| 6 | Beauty | 6 | uBetya |
| 7 | natural | 7 | a l!$% |
| 8 | festival | 8 | f'&,0$$ |
| 9 | butterfly | 9 | bttefly |
| 10 | basketball | 5 | btaale |
| 11 | Comfortable | 11 | morae |
| 12 | championship | 6 | ms3m!h |
| 13 | approximative | 13 | 2amp58"12!':; |
| 14 | communications | 7 | cot+< s |
| 15 | uncopyrightable | 15 | #& ahp= y\* 8> |
| 16 | [Macroenvironment](https://finance-ua.com.ua/slovar/macroenvironment.html) | 8 | iacornme |
| 17 | misunderstandings | 17 | 98>m:!'is;=2:;=n! |
| 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) | 9 | !6tr"/=%r |
| 19 | inappropriatenesses | 19 | p|r7 !, <> $-/at' |
| 20 | incomprehensibilitie | 10 | n!"#>%!$ + |

*Таблица 1 – Дешифрование без ключа для 100 символов*

**200 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | a | 2 | a |
| 2 | up | 2 | up |
| 3 | joy | 3 | joy |
| 4 | frog | 4 | frog |
| 5 | tiger | 5 | tiger |
| 6 | Beauty | 6 | Beauty |
| 7 | natural | 7 | natur$l |
| 8 | festival | 4 | fe ; |
| 9 | c | 9 | butzerfly |
| 10 | basketball | 5 | abaskl |
| 11 | Comfortable | 11 | Co#f\*=tabl/ |
| 12 | championship | 4 | p'.n |
| 13 | approximative | 13 | approximtive |
| 14 | communications | 14 | ommucatzins |
| 15 | uncopyrightable | 15 | 0n, 5\*&ig-&$bl+ |
| 16 | [Macroenvironment](https://finance-ua.com.ua/slovar/macroenvironment.html) | 8 | ia)!!e+1 |
| 17 | misunderstandings | 17 | (& 0n6e&sta/\*/n3s |
| 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) | 18 | chaceritclbl |
| 19 | inappropriatenesses | 19 | iappropriaenses |
| 20 | incomprehensibilitie | 3 | iee |

*Таблица 2 – Дешифрование без ключа для 200 символов*

**300 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | a | 2 | aa |
| 2 | up | 2 | up |
| 3 | joy | 3 | joy |
| 4 | frog | 3 | for |
| 5 | tiger | 5 | tiger |
| 6 | Beauty | 6 | Beauty |
| 7 | natural | 7 | n$tural |
| 8 | festival | 8 | festival |
| 9 | butterfly | 9 | butterfly |
| 10 | basketball | 10 | basketball |
| 11 | Comfortable | 11 | Comfortable |
| 12 | championship | 12 | &ha#p,on'h,5 |
| 13 | approximative | 13 | a5p7 4i$a1i7e |
| 14 | communications | 7 | cmtimonis |
| 15 | uncopyrightable | 15 | h!) +#:t $$a45) |
| 16 | [Macroenvironment](https://finance-ua.com.ua/slovar/macroenvironment.html) | 16 | Miarocrnmenv |
| 17 | misunderstandings | 17 | (is4 !e72tan!i+3; |
| 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) | 18 | &ha3$&'2ri6tic$"#< |
| 19 | inappropriatenesses | 19 | i+a$p0\*pr,$t ness s |
| 20 | incomprehensibilitie | 20 | =!i! p e,e |

*Таблица 3 – Дешифрование без ключа для 300 символов*

**400 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | a | 2 | aa |
| 2 | up | 2 | up |
| 3 | joy | 3 | joy |
| 4 | frog | 4 | frog |
| 5 | tiger | 5 | tiger |
| 6 | Beauty | 6 | Beauty |
| 7 | natural | 15 | naulatralntural |
| 8 | festival | 8 | vesalt |
| 9 | butterfly | 9 | butterfly |
| 10 | basketball | 10 | basketball |
| 11 | Comfortable | 11 | Comfortable |
| 12 | championship | 6 | cha(5/"n'&i1 |
| 13 | approximative | 13 | .<2r#1&\*2:%v# |
| 14 | communications | 14 | c\*mmcat./ |
| 15 | uncopyrightable | 15 | u'-opyri&h1ab%$ |
| 16 | [Macroenvironment](https://finance-ua.com.ua/slovar/macroenvironment.html) | 16 | $cr!7+8';o!,+"t |
| 17 | misunderstandings | 17 | 2<;"#('%& 4! |
| 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) | 18 | chrcteritclly |
| 19 | inappropriatenesses | 19 | ('7p>ro1</at$ e<&+s |
| 20 | incomprehensibilitie | 10 | !r;/7t)1#= |

*Таблица 4 – Дешифрование без ключа для 400 символов*

**500 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | a | 2 | aaa |
| 2 | up | 2 | up |
| 3 | joy | 3 | joy |
| 4 | frog | 4 | frog |
| 5 | tiger | 5 | tiger |
| 6 | Beauty | 6 | Beauty |
| 7 | natural | 7 | natural |
| 8 | festival | 8 | festival |
| 9 | butterfly | 9 | butterfly |
| 10 | basketball | 10 | basketbal |
| 11 | Comfortable | 11 | Comfortable |
| 12 | championship | 12 | championship |
| 13 | approximative | 13 | approimative |
| 14 | communications | 14 | communicatons |
| 15 | uncopyrightable | 15 | ucopyrihtabl |
| 16 | [Macroenvironment](https://finance-ua.com.ua/slovar/macroenvironment.html) | 16 | Macoroenviroment |
| 17 | misunderstandings | 17 | m,sun\*e<st(<din&s |
| 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) | 18 | chrcteristiclly |
| 19 | inappropriatenesses | 19 | inappropriateees |
| 20 | incomprehensibilitie | 20 | incomprehnsbilti |

*Таблица 5 – Дешифрование без ключа для 500 символов*

**600 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | a | 2 | aa |
| 2 | up | 2 | up |
| 3 | joy | 3 | joy |
| 4 | frog | 4 | frog |
| 5 | tiger | 5 | tiger |
| 6 | Beauty | 6 | Beauty |
| 7 | natural | 7 | natural |
| 8 | festival | 8 | festival |
| 9 | butterfly | 9 | butterfly |
| 10 | basketball | 10 | basketball |
| 11 | Comfortable | 11 | Comfortable |
| 12 | championship | 12 | championship |
| 13 | approximative | 13 | approximative |
| 14 | communications | 14 | communications |
| 15 | uncopyrightable | 15 | uncopyrghtable |
| 16 | [Macroenvironment](https://finance-ua.com.ua/slovar/macroenvironment.html) | 16 | Macroenvironment |
| 17 | misunderstandings | 17 | misundersadngs |
| 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) | 18 | characteristically |
| 19 | inappropriatenesses | 19 | iapropratenesses |
| 20 | incomprehensibilitie | 20 | igncomprehinsibilitie |

*Таблица 6 – Дешифрование без ключа для 600 символов*

**700 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | a | 2 | aa |
| 2 | up | 2 | up |
| 3 | joy | 3 | joy |
| 4 | frog | 4 | frog |
| 5 | tiger | 5 | tiger |
| 6 | Beauty | 6 | Beauty |
| 7 | natural | 7 | natural |
| 8 | festival | 8 | festival |
| 9 | butterfly | 9 | butterfly |
| 10 | basketball | 10 | basketball |
| 11 | Comfortable | 11 | Comfortable |
| 12 | championship | 12 | championship |
| 13 | approximative | 13 | approximative |
| 14 | communications | 14 | communictions |
| 15 | uncopyrightable | 15 | uncopyrightable |
| 16 | [Macroenvironment](https://finance-ua.com.ua/slovar/macroenvironment.html) | 16 | [Macroenvironment](https://finance-ua.com.ua/slovar/macroenvironment.html) |
| 17 | misunderstandings | 17 | misunderstandings |
| 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) | 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) |
| 19 | inappropriatenesses | 17 | iappopriatenesses |
| 20 | incomprehensibilitie | 20 | iapropriatenesses |

*Таблица 7 – Дешифрование без ключа для 700 символов*

**800 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | a | 3 | aa |
| 2 | up | 2 | up |
| 3 | joy | 3 | joy |
| 4 | frog | 4 | frog |
| 5 | tiger | 5 | tiger |
| 6 | Beauty | 6 | Beauty |
| 7 | natural | 7 | natural |
| 8 | festival | 8 | festival |
| 9 | butterfly | 9 | butterfly |
| 10 | basketball | 10 | basketball |
| 11 | Comfortable | 11 | Comfortable |
| 12 | championship | 12 | championship |
| 13 | approximative | 13 | approximative |
| 14 | communications | 14 | c mmun,cations |
| 15 | uncopyrightable | 15 | ncopyrightable |
| 16 | [Macroenvironment](https://finance-ua.com.ua/slovar/macroenvironment.html) | 16 | Macroenvironment |
| 17 | misunderstandings | 17 | misunderstandings |
| 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) | 18 | characteristically |
| 19 | inappropriatenesses | 19 | napproriateeses |
| 20 | incomprehensibilitie | 20 | incomprehensibilitie |

*Таблица 8 – Дешифрование без ключа для 800 символов*

**900 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | a | 2 | aa |
| 2 | up | 2 | up |
| 3 | joy | 3 | joy |
| 4 | frog | 4 | frog |
| 5 | tiger | 5 | tiger |
| 6 | Beauty | 6 | Beauty |
| 7 | natural | 7 | natural |
| 8 | festival | 8 | festival |
| 9 | butterfly | 9 | bu1terfly |
| 10 | basketball | 10 | basketball |
| 11 | Comfortable | 11 | Comfortable |
| 12 | championship | 12 | championship |
| 13 | approximative | 13 | approimative |
| 14 | communications | 14 | cmmaunications |
| 15 | uncopyrightable | 15 | uncpyrighdtable |
| 16 | [Macroenvironment](https://finance-ua.com.ua/slovar/macroenvironment.html) | 16 | Macroenvironment |
| 17 | misunderstandings | 17 | misunderstandings |
| 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) | 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) |
| 19 | inappropriatenesses | 19 | inappropriatenesses |
| 20 | incomprehensibilitie | 20 | incomprehensibilitie |

*Таблица 9 – Дешифрование без ключа для 900 символов*

**1000 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | a | 2 | aa |
| 2 | up | 2 | up |
| 3 | joy | 3 | joy |
| 4 | frog | 4 | frog |
| 5 | tiger | 5 | tiger |
| 6 | Beauty | 6 | Beauty |
| 7 | natural | 7 | natural |
| 8 | festival | 8 | festival |
| 9 | butterfly | 9 | butterfly |
| 10 | basketball | 10 | basketball |
| 11 | Comfortable | 11 | Comfortable |
| 12 | championship | 12 | championship |
| 13 | approximative | 13 | approximative |
| 14 | communications | 14 | communications |
| 15 | uncopyrightable | 15 | uncopyrightabl |
| 16 | [Macroenvironment](https://finance-ua.com.ua/slovar/macroenvironment.html) | 16 | Macroenvironment |
| 17 | misunderstandings | 17 | misunderstandings |
| 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) | 18 | characteristically |
| 19 | inappropriatenesses | 19 | inappropriatenesses |
| 20 | incomprehensibilitie | 20 | incomprehensiblitie |

*Таблица 10 – Дешифрование без ключа для 1000 символов*

**1500 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | a | 2 | aa |
| 2 | up | 2 | up |
| 3 | joy | 3 | joy |
| 4 | frog | 4 | frog |
| 5 | tiger | 5 | tiger |
| 6 | Beauty | 6 | Beauty |
| 7 | natural | 7 | natural |
| 8 | festival | 8 | festival |
| 9 | butterfly | 9 | butterfly |
| 10 | basketball | 10 | basketball |
| 11 | Comfortable | 11 | Comfortable |
| 12 | championship | 12 | championship |
| 13 | approximative | 13 | approximative |
| 14 | communications | 14 | communications |
| 15 | uncopyrightable | 15 | uncopyrightable |
| 16 | [Macroenvironment](https://finance-ua.com.ua/slovar/macroenvironment.html) | 16 | Macrenvironment |
| 17 | misunderstandings | 17 | msunderstandings |
| 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) | 18 | characteristically |
| 19 | inappropriatenesses | 19 | inappropriatenesses |
| 20 | incomprehensibilitie | 20 | incomprehensibilitie |

*Таблица 11 – Дешифрование без ключа для 1500 символов*

**3000 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | a | 2 | aa |
| 2 | up | 2 | up |
| 3 | joy | 3 | joy |
| 4 | frog | 4 | frog |
| 5 | tiger | 5 | tiger |
| 6 | Beauty | 6 | Beauty |
| 7 | natural | 7 | natural |
| 8 | festival | 8 | festival |
| 9 | butterfly | 9 | butterfly |
| 10 | basketball | 10 | basketball |
| 11 | Comfortable | 11 | Comfortable |
| 12 | championship | 12 | championship |
| 13 | approximative | 13 | approximative |
| 14 | communications | 14 | communications |
| 15 | uncopyrightable | 15 | uncopyrightable |
| 16 | [Macroenvironment](https://finance-ua.com.ua/slovar/macroenvironment.html) | 16 | Macroenvironment |
| 17 | misunderstandings | 17 | misunderstandings |
| 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) | 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) |
| 19 | inappropriatenesses | 19 | inappropriatenesses |
| 20 | incomprehensibilitie | 20 | incomprehensibilitie |

*Таблица 12 – Дешифрование без ключа для 3000 символов*

**6000 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | a | 2 | aa |
| 2 | up | 2 | up |
| 3 | joy | 3 | joy |
| 4 | frog | 4 | frog |
| 5 | tiger | 5 | tiger |
| 6 | Beauty | 6 | Beauty |
| 7 | natural | 7 | natural |
| 8 | festival | 8 | festival |
| 9 | butterfly | 9 | butterfly |
| 10 | basketball | 10 | basketball |
| 11 | Comfortable | 11 | Comfortable |
| 12 | championship | 12 | championship |
| 13 | approximative | 13 | approximative |
| 14 | communications | 14 | communications |
| 15 | uncopyrightable | 15 | uncopyrightable |
| 16 | [Macroenvironment](https://finance-ua.com.ua/slovar/macroenvironment.html) | 16 | [Macroenvironment](https://finance-ua.com.ua/slovar/macroenvironment.html) |
| 17 | misunderstandings | 17 | misunderstandings |
| 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) | 18 | [characteristically](https://ru.wiktionary.org/wiki/characteristically) |
| 19 | inappropriatenesses | 19 | inappropriatenesses |
| 20 | incomprehensibilitie | 20 | incomprehensibilitie |

*Таблица 13 – Дешифрование без ключа для 6000 символов*

***Русские символы:***

**100 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | а | 1 | а |
| 2 | Ау | 1 | у |
| 3 | кот | 2 | кт |
| 4 | панк | 1 | а |
| 5 | дурак | 1 | а |
| 6 | авария | 1 | р |
| 7 | Бабушка | 3 | Бак |
| 8 | багряный | 3 | анй |
| 9 | авторитет | 1 | т |
| 10 | Архитектор | 5 | оррще |
| 11 | гастрономия | 2 | тя |
| 12 | авиаракетный | 1 | н |
| 13 | автобиография | 4 | тбри |
| 14 | абстракционизм | 2 | ба |
| 15 | аристократичный | 3 | кбы |
| 16 | благодарственный | 5 | дтонй |
| 17 | водонепроницаемый | 6 | нпваеы |
| 18 | градостроительный | 6 | отвльы |
| 19 | благотворительность | 9 | агвотильо |
| 20 | золотопромышленность | 1 | о |

*Таблица 14 – Дешифрование без ключа для 100 символов*

**200 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | а | 1 | а |
| 2 | Ау | 2 | Ау |
| 3 | кот | 3 | кот |
| 4 | панк | 1 | а |
| 5 | дурак | 4 | драк |
| 6 | авария | 5 | вария |
| 7 | Бабушка | 4 | Баук |
| 8 | багряный | 5 | агянй |
| 9 | авторитет | 1 | т |
| 10 | Архитектор | 6 | Аретор |
| 11 | гастрономия | 5 | стрндя |
| 12 | авиаракетный | 8 | вааелтны |
| 13 | автобиография | 7 | тбиогра |
| 14 | абстракционизм | 7 | амбацон |
| 15 | аристократичный | 10 | райаитокый |
| 16 | благодарственный | 6 | ладтнй |
| 17 | водонепроницаемый | 8 | оононцей |
| 18 | градостроительный | 8 | рдооиеьй |
| 19 | благотворительность | 8 | баооильт |
| 20 | золотопромышленность | 7 | онсротм |

*Таблица 15 – Дешифрование без ключа для 200 символов*

**300 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | а | 1 | а |
| 2 | Ау | 2 | Ау |
| 3 | кот | 3 | кот |
| 4 | панк | 4 | панк |
| 5 | дурак | 5 | дурак |
| 6 | авария | 7 | вряария |
| 7 | Бабушка | 5 | Бабук |
| 8 | багряный | 6 | агрянй |
| 9 | авторитет | 8 | автоитет |
| 10 | Архитектор | 6 | ртетор |
| 11 | гастрономия | 5 | астня |
| 12 | авиаракетный | 7 | ваатный |
| 13 | автобиография | 6 | тбигра |
| 14 | абстракционизм | 8 | амбацони |
| 15 | аристократичный | 9 | аисратчый |
| 16 | благодарственный | 9 | бладтеный |
| 17 | водонепроницаемый | 9 | оонеонцей |
| 18 | градостроительный | 9 | рдосоиеьй |
| 19 | благотворительность | 11 | багооильост |
| 20 | золотопромышленность | 11 | ооторомнст |

*Таблица 16 – Дешифрование без ключа для 300 символов*

**400 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | а | 1 | а |
| 2 | Ау | 2 | Ау |
| 3 | кот | 3 | кот |
| 4 | панк | 4 | панк |
| 5 | дурак | 5 | дурак |
| 6 | авария | 6 | авария |
| 7 | Бабушка | 14 | БабшкБашкБаука |
| 8 | багряный | 6 | барянй |
| 9 | авторитет | 5 | авоет |
| 10 | Архитектор | 7 | Ариетор |
| 11 | гастрономия | 6 | гстнмя |
| 12 | авиаракетный | 9 | авараетнй |
| 13 | автобиография | 6 | тбиора |
| 14 | абстракционизм | 8 | бацонизм |
| 15 | аристократичный | 10 | аисоатичый |
| 16 | благодарственный | 9 | лдрстеный |
| 17 | водонепроницаемый | 9 | воононцей |
| 18 | градостроительный | 9 | грдооиеьй |
| 19 | благотворительность | 9 | баоиелост |
| 20 | золотопромышленность | 12 | зооторомшнст |

*Таблица 17 – Дешифрование без ключа для 400 символов*

**500 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | а | 1 | а |
| 2 | Ау | 2 | Ау |
| 3 | кот | 3 | кот |
| 4 | панк | 4 | панк |
| 5 | дурак | 5 | дурак |
| 6 | авария | 6 | авария |
| 7 | Бабушка | 7 | Бабушка |
| 8 | багряный | 8 | багряный |
| 9 | авторитет | 7 | автотет |
| 10 | Архитектор | 9 | Аритектор |
| 11 | гастрономия | 7 | гстномя |
| 12 | авиаракетный | 12 | аваркетниаан |
| 13 | автобиография | 8 | атбиораи |
| 14 | абстракционизм | 10 | абационизм |
| 15 | аристократичный | 12 | арисоатичный |
| 16 | благодарственный | 10 | блодртеный |
| 17 | водонепроницаемый | 10 | воонпонцей |
| 18 | градостроительный | 10 | грдотоиеьй |
| 19 | благотворительность | 10 | баооиелост |
| 20 | золотопромышленность | 12 | зоопромыност |

*Таблица 18 – Дешифрование без ключа для 500 символов*

**600 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | а | 1 | а |
| 2 | Ау | 2 | Ау |
| 3 | кот | 3 | кот |
| 4 | панк | 4 | панк |
| 5 | дурак | 5 | дурак |
| 6 | авария | 6 | авария |
| 7 | Бабушка | 7 | Бабушка |
| 8 | багряный | 8 | багряный |
| 9 | авторитет | 9 | авторитет |
| 10 | Архитектор | 10 | Архитектор |
| 11 | гастрономия | 11 | гастрономия |
| 12 | авиаракетный | 12 | авиаракетный |
| 13 | автобиография | 8 | тбиограи |
| 14 | абстракционизм | 10 | абационизм |
| 15 | аристократичный | 11 | аристотичнй |
| 16 | благодарственный | 12 | лгодрственый |
| 17 | водонепроницаемый | 10 | оонпонцаей |
| 18 | градостроительный | 10 | рдотоиельй |
| 19 | благотворительность | 13 | блаотоиелость |
| 20 | золотопромышленность | 13 | зоопромыность |

*Таблица 19 – Дешифрование без ключа для 600 символов*

**700 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | а | 1 | а |
| 2 | Ау | 4 | АуАу |
| 3 | кот | 3 | кот |
| 4 | панк | 4 | панк |
| 5 | дурак | 5 | дурак |
| 6 | авария | 6 | авария |
| 7 | Бабушка | 7 | Бабушка |
| 8 | багряный | 8 | багряный |
| 9 | авторитет | 9 | авторитет |
| 10 | Архитектор | 10 | Архитектор |
| 11 | гастрономия | 11 | гастрономия |
| 12 | авиаракетный | 12 | авиаракетный |
| 13 | автобиография | 13 | автобиография |
| 14 | абстракционизм | 10 | абационизм |
| 15 | аристократичный | 13 | арстокатичный |
| 16 | благодарственный | 12 | блгдрственны |
| 17 | водонепроницаемый | 11 | однепонцаей |
| 18 | градостроительный | 11 | раостоиельй |
| 19 | благотворительность | 11 | баготоиеост |
| 20 | золотопромышленность | 14 | зооопомышность |

*Таблица 20 – Дешифрование без ключа для 700 символов*

**800 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | а | 1 | а |
| 2 | Ау | 4 | АуАу |
| 3 | кот | 3 | кот |
| 4 | панк | 4 | панк |
| 5 | дурак | 5 | дурак |
| 6 | авария | 6 | авария |
| 7 | Бабушка | 7 | Бабушка |
| 8 | багряный | 8 | багряный |
| 9 | авторитет | 9 | авторитет |
| 10 | Архитектор | 10 | Арихитектор |
| 11 | гастрономия | 11 | гастрономия |
| 12 | авиаракетный | 12 | авиаракетный |
| 13 | автобиография | 13 | автобиография |
| 14 | абстракционизм | 14 | абстракционизм |
| 15 | аристократичный | 15 | аристократичный |
| 16 | благодарственный | 14 | благдрственный |
| 17 | водонепроницаемый | 11 | однепонцаей |
| 18 | градостроительный | 11 | раостоиельй |
| 19 | благотворительность | 13 | благотоиеьост |
| 20 | золотопромышленность | 13 | зоотопромышнс |

*Таблица 21 – Дешифрование без ключа для 800 символов*

**900 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | а | 2 | аа |
| 2 | Ау | 2 | Ау |
| 3 | кот | 3 | кот |
| 4 | панк | 4 | панк |
| 5 | дурак | 5 | дурак |
| 6 | авария | 6 | авария |
| 7 | Бабушка | 7 | Бабушка |
| 8 | багряный | 8 | багряный |
| 9 | авторитет | 9 | авторитет |
| 10 | Архитектор | 10 | Архитектор |
| 11 | гастрономия | 11 | гастрономия |
| 12 | авиаракетный | 12 | авиаракетный |
| 13 | автобиография | 13 | автобиография |
| 14 | абстракционизм | 14 | абстракционизм |
| 15 | аристократичный | 15 | аристократичный |
| 16 | благодарственный | 16 | благодарственный |
| 17 | водонепроницаемый | 17 | водонепроницаемый |
| 18 | градостроительный | 13 | градостоиельй |
| 19 | благотворительность | 14 | благотоителост |
| 20 | золотопромышленность | 15 | зоотопромышнось |

*Таблица 22 – Дешифрование без ключа для 900 символов*

**1000 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | а | 3 | ааа |
| 2 | Ау | 4 | АуАу |
| 3 | кот | 3 | кот |
| 4 | панк | 4 | панк |
| 5 | дурак | 5 | дурак |
| 6 | авария | 6 | авария |
| 7 | Бабушка | 7 | Бабушка |
| 8 | багряный | 8 | багряный |
| 9 | авторитет | 9 | авторитет |
| 10 | Архтектор | 10 | Архитектор |
| 11 | гастрономия | 11 | гастрономия |
| 12 | авиаракетный | 12 | авиаракетный |
| 13 | автобиография | 13 | автобиография |
| 14 | абстракционизм | 14 | абстракционизм |
| 15 | аристократичный | 15 | аристократичный |
| 16 | благодарственный | 16 | благодарственный |
| 17 | водонепроницаемый | 17 | водонепроницаемый |
| 18 | градостроительный | 18 | градостроительный |
| 19 | благотворительность | 19 | благотворительность |
| 20 | золотопромышленность | 20 | золотопромышленность |

*Таблица 23 – Дешифрование без ключа для 1000 символов*

**1500 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | а | 3 | ааа |
| 2 | Ау | 4 | АуАу |
| 3 | кот | 3 | кот |
| 4 | панк | 4 | панк |
| 5 | дурак | 5 | дурак |
| 6 | авария | 6 | авария |
| 7 | Бабушка | 7 | Бабушка |
| 8 | багряный | 8 | багряный |
| 9 | авторитет | 9 | авторитет |
| 10 | Архитектор | 9 | Архитектр |
| 11 | гастрономия | 10 | гастрномия |
| 12 | авиаракетный | 11 | виаракетный |
| 13 | автобиография | 12 | автобиорафия |
| 14 | абстракционизм | 13 | абстракционим |
| 15 | аристократичный | 12 | аритокраиный |
| 16 | благодарственный | 15 | благодарственны |
| 17 | водонепроницаемый | 16 | водонепрониаемый |
| 18 | градостроительный | 17 | градостроитльный |
| 19 | благотворительность | 16 | бготворитльность |
| 20 | золотопромышленность | 17 | золотопрмышленось |

*Таблица 24 – Дешифрование без ключа для 1500 символов*

**3000 символов:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | а | 2 | аа |
| 2 | Ау | 2 | Ау |
| 3 | кот | 3 | кот |
| 4 | панк | 4 | панк |
| 5 | дурак | 5 | дурак |
| 6 | авария | 6 | авария |
| 7 | Бабушка | 7 | Бабушка |
| 8 | багряный | 8 | багряный |
| 9 | авторитет | 9 | авторитет |
| 10 | Архитектор | 10 | Архитектор |
| 11 | гастрономия | 11 | гастрономия |
| 12 | авиаракетный | 12 | авиаракетный |
| 13 | автобиография | 11 | авобиогрфия |
| 14 | абстракционизм | 14 | абстракционизм |
| 15 | аристократичный | 15 | аристократичный |
| 16 | благодарственный | 15 | благодарстенный |
| 17 | водонепроницаемый | 17 | водонепроницаемый |
| 18 | градостроительный | 18 | градостроительный |
| 19 | благотворительность | 18 | благотворительност |
| 20 | золотопромышленность | 20 | золотопромышленность |

*Таблица 25 – Дешифрование без ключа для 3000 символов*

**6000 символов:**

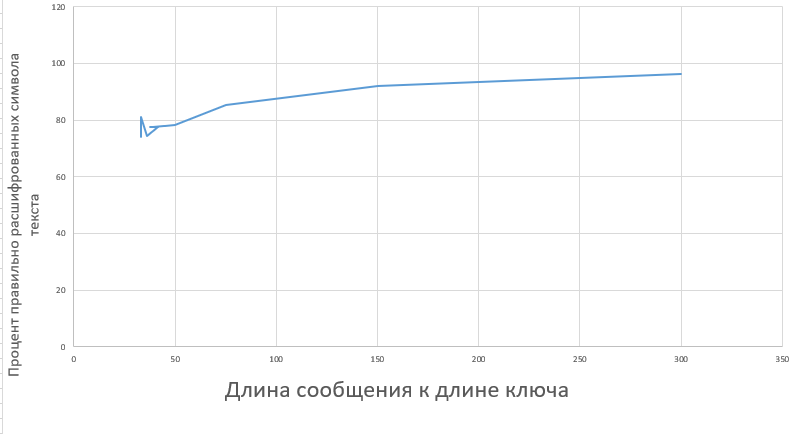
|  |  |  |  |
| --- | --- | --- | --- |
| **Длина ключа** | **Ключ** | **Длина ключа, полученного программой** | **Ключ, полученный программой** |
| 1 | а | 2 | аа |
| 2 | Ау | 2 | Ау |
| 3 | кот | 3 | кот |
| 4 | панк | 4 | панк |
| 5 | дурак | 5 | дурак |
| 6 | авария | 6 | авария |
| 7 | Бабушка | 7 | Бабушка |
| 8 | багряный | 8 | багряный |
| 9 | авторитет | 9 | авторитет |
| 10 | Архитектор | 10 | Архитектор |
| 11 | гастрономия | 11 | гастрономия |
| 12 | авиаракетный | 12 | авиаракетный |
| 13 | автобиография | 13 | автобиография |
| 14 | абстракционизм | 14 | абстракционизм |
| 15 | аристократичный | 15 | аристократичный |
| 16 | благодарственный | 16 | благодарственный |
| 17 | водонепроницаемый | 17 | водонепроницаемый |
| 18 | градостроительный | 18 | градостроительный |
| 19 | благотворительность | 19 | благотворительность |
| 20 | золотопромышленность | 20 | золотопромышленность |

*Таблица 26 – Дешифрование без ключа для 6000 символов*

*Анализ результатов*

|  |  |  |  |
| --- | --- | --- | --- |
| ***Длина ключа*** | ***Процент правильно дешифрованных символов по длине ключа для английских символов*** | ***Длина ключа*** | ***Процент правильно дешифрованных символов по длине ключа для русских символов*** |
| ***1*** | 1 | ***1*** | 1 |
| ***2*** | 1 | ***2*** | 0,961 |
| ***3*** | 1 | ***3*** | 0,974 |
| ***4*** | 0,942 | ***4*** | 0,884 |
| ***5*** | 0,923 | ***5*** | 0,923 |
| ***6*** | 1 | ***6*** | 0,910 |
| ***7*** | 0,912 | ***7*** | 0,89 |
| ***8*** | 0,856 | ***8*** | 0,875 |
| ***9*** | 0,974 | ***9*** | 0,743 |
| ***10*** | 0,931 | ***10*** | 0,761 |
| ***11*** | 0,923 | ***11*** | 0,741 |
| ***12*** | 0,801 | ***12*** | 0,82 |
| ***13*** | 0,817 | ***13*** | 0,715 |
| ***14*** | 0,841 | ***14*** | 0,681 |
| ***15*** | 0,764 | ***15*** | 0,753 |
| ***16*** | 0,817 | ***16*** | 0,716 |
| ***17*** | 0,719 | ***17*** | 0,678 |
| ***18*** | 0,821 | ***18*** | 0,653 |
| ***19*** | 0,785 | ***19*** | 0,680 |
| ***20*** | 0,719 | ***20*** | 0,654 |

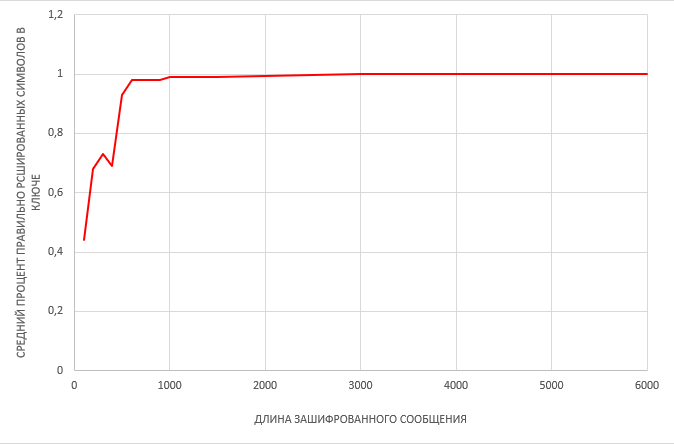
*Таблица 29 – Среднее значение процента правильно дешифрованного количества символов для разных длин ключа (английские и русские символы)*



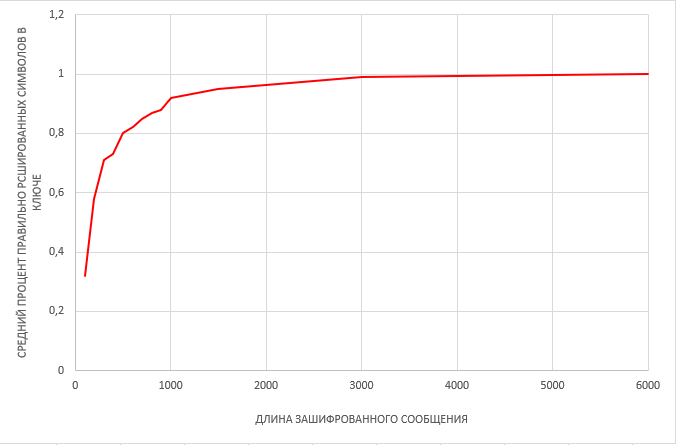
*График 1 - Отношение процента правильно расшифрованных символов сообщения к пропорции длина сообщения к длине ключа для английских символов*

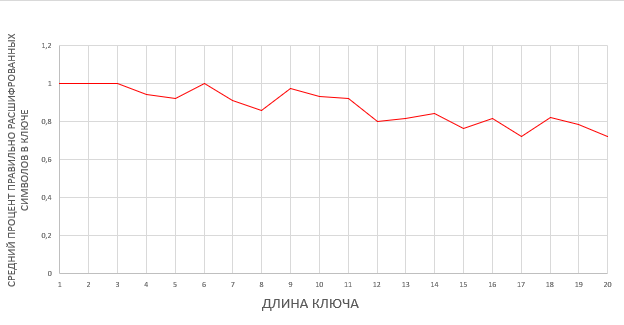
**

*График 2 - Отношение процента правильно расшифрованных символов сообщения к пропорции длина сообщения к длине ключа для русских символов*

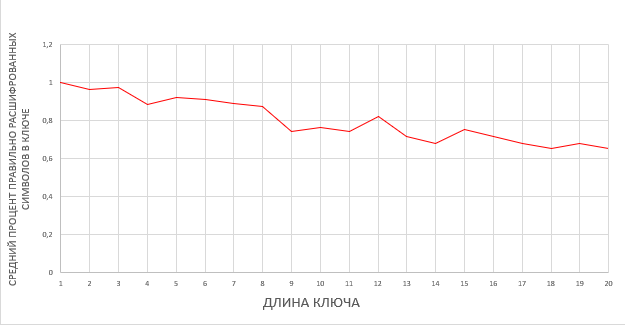
******

*График 3 – Зависимость вероятности расшифрованных символов ключа от длины зашифрованного сообщения для английских символов*

****** *График 4 – Зависимость вероятности расшифрованных символов ключа от длины зашифрованного сообщения для русских символов*

******

*График 5 – Зависимость вероятности правильно расшифрованных символов в ключе от длины ключа для английских символов*

**

*График 6 – Зависимость вероятности правильно расшифрованных символов в ключе от длины ключа для русских символов*

Из графиков, заметно, что для английских символов в среднем для дешифровки ключа требуется длина текста меньше, чем для русских. Возможно, это связано с особенностью кодировки русских символов, а также с тем, что в русском алфавите больше символов, чем в английском языке.

## Выводы

В ходе данной работы мы командой реализовали программу, которая шифрует исходный текст с ключом при помощи XOR и дешифрует зашифрованное сообщение без ключа при помощи частотного анализа для русских и английских символов. В результате тестирования программы и анализа результатов мы выявили, что данная программа лучше всего работает для зашифрованных текстов, длина которых больше 600 символов, особенно для английских символов. Из графиков видно, что шанс дешифровать текст на английском без ключа выше, чем для русских символов, что можно объяснить следующим: Метод частотного анализа основывается на анализе частоты встречаемости символов в тексте. В английском языке встречаются относительно небольшое количество символов (только 26 букв латинского алфавита и несколько знаков препинания), поэтому распределение частот встречаемости символов в английском тексте имеет характерные черты, которые можно узнать и использовать для дешифровки. Однако в русском языке используется значительно большее количество символов (33 буквы кириллицы, различные знаки препинания и дополнительные знаки), что делает распределение частот встречаемости символов более равномерным и менее характерным. Это затрудняет процесс дешифровки при использовании метода частотного анализа. Не стоит забывать и про то, что для русских и английских символов используются разные кодировочные таблицы: UTF-8 для английских и Windows-1251 для русских, что также может повлечь проблемы при дешифровке. Таким образом, различия в распределении символов в английском и русском тексте могут приводить к различной эффективности частотного анализа в XOR-шифровании для этих языков.

## *Приложение*

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <malloc.h>

#include <string.h>

#include <stdlib.h>

#include <locale.h>

#include <time.h>

#include <math.h>

#include <ctype.h>

#define BLOCK\_SIZE 5

#define MAX\_SIZE 30

#define PROBABILITY 0.046

#define GENERATION 5

#define ALPHABET\_EN "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz"

#define ALPHABET\_RU "АБВГДЕЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯабвгдежзийклмнопрстуфхцчшщъыьэюяЁё"

#define SPECIAL "123456789.,?!:;()[]{}~#№^\"\'-–—«»„“”‘’‛‟`…‹›<>‒•·&@#$%\*+=\_/\| \n"

enum { END = 0, CODE = 1, DECODE\_KEY = 2, DECODE\_NOKEY = 3, EXIT = 4 };

unsigned char mbkey[1024] = { 0 };

int p;

typedef struct

{

unsigned char File;

unsigned char Key;

unsigned char Text\_Generate;

} Symbol;

//СТРУКТУРА ФАЙЛОВ//

typedef struct

{

FILE\* Start;

FILE\* End;

FILE\* Coded;

FILE\* Key;

FILE\* Generate\_Text;

FILE\* Probably\_Keys;

} File;

//

void insertionSort(int arr1[], int arr2[], int n)

{

int i, key, j, key1;

for (i = 1; i < n; i++)

{

key = arr1[i];

key1 = arr2[i];

j = i - 1;

// Перемещаем элементы arr[0..i-1], которые больше key, на одну позицию вперед

// относительно текущей позиции j

while (j >= 0 && arr1[j] < key)

{

arr1[j + 1] = arr1[j];

arr2[j+1] = arr2[j+1];

j = j - 1;

}

arr1[j + 1] = key;

arr2[j + 1] = key1;

}

}

void removeSpaces(unsigned char\* str)

{

int i, j;

for (i = 0, j = 0; str[i] != '\0'; i++)

{

if (str[i] != ' ')

{

str[j++] = str[i];

}

}

str[j] = '\0';

}

//КОДИРОВАНИЕ//

void Coding()

{

Symbol Sym;

File F;

F.Start = fopen("Start.txt", "rb");;//исходный текст

F.Coded = fopen("Coded.txt", "wb");

F.Key = fopen("Key.txt", "rb");

Sym.File = '\0', Sym.Key = '\0';

fseek(F.Start, 0, SEEK\_END);

int buf1 = ftell(F.Start);//ищем длину исходного текста

fseek(F.Start, 0, SEEK\_SET);

fseek(F.Key, 0, SEEK\_END);

int buf2 = ftell(F.Key);//ищем длину ключа

fseek(F.Key, 0, SEEK\_SET);

while (ftell(F.Start) != buf1)

{

Sym.File = fgetc(F.Start);

if (ftell(F.Key) < buf2)

Sym.Key = fgetc(F.Key);

else

{

fseek(F.Key, 0, SEEK\_SET);

Sym.Key = fgetc(F.Key);

}

Sym.File ^= Sym.Key;//посимвольное шифрование

fputc(Sym.File, F.Coded);

}

fclose(F.Coded);

fclose(F.Start);

fclose(F.Key);

}

//ДЕШИФРОВАНИЕ С КЛЮЧОМ//

void Decoding\_Key()

{

Symbol Sym;

File F;

F.End = fopen("End.txt", "wb");

F.Coded = fopen("Coded.txt", "rb");

F.Key = fopen("Key.txt", "rb");

Sym.File = '\0', Sym.Key = '\0';

fseek(F.Coded, 0, SEEK\_END);

int buf1 = ftell(F.Coded);//ищем длину исходного текста

fseek(F.Coded, 0, SEEK\_SET);

fseek(F.Key, 0, SEEK\_END);

int buf2 = ftell(F.Key);//ищем длину ключа

fseek(F.Key, 0, SEEK\_SET);

while (ftell(F.Coded) != buf1)

{

Sym.File = fgetc(F.Coded);

if (ftell(F.Key) < buf2)

Sym.Key = fgetc(F.Key);

else

{

fseek(F.Key, 0, SEEK\_SET);

Sym.Key = fgetc(F.Key);

}

Sym.File ^= Sym.Key;//посимвольное дешифрование

fputc(Sym.File, F.End);

}

fclose(F.End);

fclose(F.Coded);

fclose(F.Key);

}

//ИНДЕКС СОВПАДЕНИЯ//

float coincidence\_index(int len, unsigned char\* block)

{

long long freq[256] = { 0 }; // массив для хранения частот символов

float coincidence = 0.0;//совпадение

for (int i = 0; i < len; i++)

{

freq[block[i]]++;

}

for (int i = 0; i < 256; i++)

{

coincidence += freq[i] \* (freq[i] - 1);

}

coincidence /= len \* (len - 1);

return coincidence;

}

//

void print\_pattern\_chars(FILE\* output, unsigned char\* str, int distance)

{

int length = strlen(str);

int i;

for (i = 0; i < length - distance + 1; i += distance)

{

int j;

for (j = i; j < i + distance; j++)

{

printf("%c", str[j]);

fprintf(output, "%c", str[j]);

}

printf("\n");

fprintf(output, "%c", '\n');

}

}

//НАХОЖДЕНИЕ ДЛИНЫ КЛЮЧА//

int LenKey()

{

Symbol Sym;

File F;

F.Coded = fopen("Coded.txt", "rb");

Sym.File = '\0', Sym.Key = '\0';

fseek(F.Coded, 0, SEEK\_END);

int buf1 = ftell(F.Coded);//длина зашифрованного текста

fseek(F.Coded, 0, SEEK\_SET);

float\* coincidence\_index\_avg;//динамеческий массив для индексов совпадения

coincidence\_index\_avg = (float\*)malloc(MAX\_SIZE \* sizeof(float));

memset(coincidence\_index\_avg, 0, MAX\_SIZE);

for (int key\_len = 1; key\_len <= MAX\_SIZE; key\_len++)//перебор длины ключа

{

int count = 0;//для подсчета количества блоков

coincidence\_index\_avg[key\_len - 1] = 0.0;//перебор различного сдвига

for (int i = 0; i < key\_len; i++)

{

unsigned char block[BLOCK\_SIZE] = { 0 };//массив для хранения блока

int block\_count = 0;//счетчик блоков

for (int j = i; j < buf1; j += key\_len)

{

fseek(F.Coded, j, SEEK\_SET);

block[block\_count++] = fgetc(F.Coded);

if (block\_count == BLOCK\_SIZE)

{

coincidence\_index\_avg[key\_len - 1] += coincidence\_index(BLOCK\_SIZE, block);

count++;

block\_count = 0;

}

}

rewind(F.Coded);

}

coincidence\_index\_avg[key\_len - 1] /= count;//среднее количество совпадений символов

printf("Длина %d: %.3lf\n", key\_len, coincidence\_index\_avg[key\_len - 1]);

}

int MaxKeyLength = 1;

float max\_coincidence\_index = 0.0;

for (int i = 1; i < MAX\_SIZE; i++)

{

if (coincidence\_index\_avg[i] >= PROBABILITY)

{

max\_coincidence\_index = coincidence\_index\_avg[i];

MaxKeyLength = i + 1;

break;

}

}

fclose(F.Coded);

return MaxKeyLength;

}

//НАХОЖДЕНИЕ КЛЮЧА//

void FindKey(int Length\_Key)

{

Symbol Sym;

File F;

F.Coded = fopen("Coded.txt", "rb");

Sym.File = '\0', Sym.Key = '\0';

F.Probably\_Keys = fopen("probably\_keys.txt", "wb");

fseek(F.Coded, 0, SEEK\_END);

int buf1 = ftell(F.Coded);//длина зашифрованного текста

fseek(F.Coded, 0, SEEK\_SET);

p = 0;

for (int i = 0; i < Length\_Key; i++)

{

unsigned char\* block;//для анализирования количества символов

block = (unsigned char\*)malloc((buf1 / Length\_Key + 1) \* sizeof(unsigned char));

memset(block, 0, (buf1 / Length\_Key + 1) \* sizeof(unsigned char));

int index = 0;

for (int j = i; j < buf1; j += Length\_Key)

{

fseek(F.Coded, j, SEEK\_SET);

block[index] = fgetc(F.Coded);

block[index] ^= ' ';//xor c самым популярным символом

if (block[index] == 0)

block[index] = '0';

index++;

}

rewind(F.Coded);

block[index] = '\0';

unsigned char most\_common = 0;//самый популярный символ зашифрованного текста

int most\_count = 0;//его индекс

int j = 0;

int Kol[256] = { 0 };

int Alphabet[256] = {0};

for (int t = 0; t < 256; t++) Alphabet[t] = t;

while (j < 256)

{

int count = 0;

for (int k = 0; k < strlen(block); k++)

{

if (block[k] == j) Kol[j] += 1;

}

j++;

}

insertionSort(Kol, Alphabet, 256);

for (int m = 0; m < GENERATION; m++)

{

mbkey[p++] = Alphabet[m];

if (m == 0)

fputc(Alphabet[m], F.Probably\_Keys);

}

}

fputc('\n', F.Probably\_Keys);

fclose(F.Coded);

fclose(F.Probably\_Keys);

}

//ДЕШИФРОВАНИЕ БЕЗ КЛЮЧА//

void Decoding\_NoKey()

{

Symbol Sym;

File F;

F.Coded = fopen("Coded.txt", "rb");//Открываем закодированное сообщение

Sym.File = '\0', Sym.Key = '\0';

fseek(F.Coded, 0, SEEK\_END);

int buf1 = ftell(F.Coded);//узнаем длину закодированного сообщения

fseek(F.Coded, 0, SEEK\_SET);

int Length\_Key = 0;

Length\_Key = LenKey();//длина ключа

printf("Длина ключа = %d\n", Length\_Key);

FindKey(Length\_Key);//находим ключ

mbkey[p] = '\0';

int flag = 0;

FILE\* output;

output = fopen("probably\_keys.txt", "wb");

printf("\nВозможные ключи:\n");

int m = 0;

for (int x = 0; x < GENERATION - 1; x++)

{

m = x;

while(m < strlen(mbkey))

{

printf("%c", mbkey[m]);

fputc(mbkey[m], output);

m += GENERATION;

}

printf("\n");

fputc('\n', output);

}

m = 0;

for (m = 0; m < p; m++)

{

if ((strchr(ALPHABET\_EN, mbkey[m]) == NULL) && (strchr(ALPHABET\_RU, mbkey[m]) == NULL) && (strchr(SPECIAL, mbkey[m]) == NULL))

{

mbkey[m] = ' ';

}

}

print\_pattern\_chars(output, mbkey , Length\_Key);

m = 0;

int v = 0;

char mb\_ru[31] = { 0 };

char mb\_en[31] = { 0 };

while (m < strlen(mbkey))

{

if (strchr(ALPHABET\_EN, mbkey[m]) != NULL)

{

mb\_en[v++] = mbkey[m];

}

m++;

}

mb\_en[v] = '\0';

fprintf(output, "%s\n", mb\_en);

printf("%s\n", mb\_en);

m = 0;

v = 0;

while (m < strlen(mbkey))

{

if (strchr(ALPHABET\_RU, mbkey[m]) != NULL)

{

mb\_ru[v++] = mbkey[m];

}

m++;

}

mb\_ru[v] = '\0';

fprintf(output, "%s\n", mb\_ru);

printf("%s\n", mb\_ru);

if (flag = 1) removeSpaces(mbkey);

mbkey[Length\_Key] = '\0';

printf("%s\n", mbkey);

fprintf(output, "%s\n", mbkey);

fclose(output);

F.Probably\_Keys = fopen("probably\_keys.txt", "rb");

fseek(F.Probably\_Keys, 0, SEEK\_END);

int Size = ftell(F.Probably\_Keys);

rewind(F.Probably\_Keys);

unsigned char Final\_Key[MAX\_SIZE + 1] = { 0 };

double MAX\_Percent = 0.0;

while (ftell(F.Probably\_Keys) != Size)

{

unsigned char s[MAX\_SIZE + 1] = { 0 };

fgets(s, 30, F.Probably\_Keys);

F.Generate\_Text = fopen("Generate\_TEXT.txt", "wb");

int count = 0;

double Percent = 0.0;

int n = 0;

while (ftell(F.Coded) != buf1)

{

Sym.File = fgetc(F.Coded);

if (n < strlen(s) && s[n] != '\n')

Sym.Key = s[n++];

else

{

n = 0;

Sym.Key = s[n++];

}

Sym.File ^= Sym.Key;//посимвольное дешифрование

if ((strchr(ALPHABET\_EN, Sym.File) != NULL) || (strchr(ALPHABET\_RU, Sym.File) != NULL))

count++;

fputc(Sym.File, F.Generate\_Text);

}

rewind(F.Coded);

Percent = (double)count / (double)buf1;

if (Percent >= MAX\_Percent)

{

MAX\_Percent = Percent;

memset(Final\_Key, 0, MAX\_SIZE + 1);

if (s[strlen(s) - 1] = '\n') s[strlen(s) - 1] = '\0';

strcpy(Final\_Key, s);

}

fclose(F.Generate\_Text);

}

printf("\nИтоговый ключ: %s", Final\_Key);

F.Key = fopen("Generate\_KEY.txt", "wb");

fputs(Final\_Key,F.Key);

fclose(F.Key);

F.Key = fopen("Generate\_KEY.txt", "rb");

fseek(F.Key, 0, SEEK\_END);

int buf2 = ftell(F.Key);

rewind(F.Key);

F.Generate\_Text = fopen("Generate\_TEXT.txt", "wb");

while (ftell(F.Coded) != buf1)

{

Sym.File = fgetc(F.Coded);

if (ftell(F.Key) < buf2)

Sym.Key = fgetc(F.Key);

else

{

fseek(F.Key, 0, SEEK\_SET);

Sym.Key = fgetc(F.Key);

}

Sym.File ^= Sym.Key;//посимвольное дешифрование

fputc(Sym.File, F.Generate\_Text);

}

fclose(F.Coded);

fclose(F.Generate\_Text);

fclose(F.Key);

fclose(F.Probably\_Keys);

memset(mbkey, 0, 1024);

printf("\n");

system("pause");

system("cls");

}

void Menu()

{

int Select;

while (1)

{

system("cls");

printf("------------------------------------\n");

printf("| |\n");

printf("| Что вы выберете? |\n");

printf("| |\n");

printf("| 1) Шифрование |\n");

printf("| |\n");

printf("| 2) Дешифрование с ключом |\n");

printf("| |\n");

printf("| 3) Дешифрование без ключа |\n");

printf("| |\n");

printf("| 4) Выход |\n");

printf("| |\n");

printf("------------------------------------\n");

printf("Ввод: ");

scanf\_s("%d", &Select);

while (getchar() != '\n');

system("cls");

if (Select == CODE)

{

Coding();

continue;

}

else

if (Select == DECODE\_KEY)

{

Decoding\_Key();

continue;

}

else

if (Select == DECODE\_NOKEY)

{

Decoding\_NoKey();

continue;

}

else

if (Select == EXIT)

{

return EXIT;

}

else

{

printf("Неверное значение! Введите снова\n");

Sleep(1000);

system("cls");

}

}

}

int main()

{

setlocale(LC\_ALL, "ru");

Menu();

return END;

}