Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Курганский государственный университет» (КГУ)

*Кафедра информационной безопасности автоматизированных систем*

**РАЗРАБОТКА ПРИЛОЖЕНИЯ  
«Разработка программных средств администрирования ОС Windows:** **Получение списка информационных ресурсов к которым имеет доступ на чтение задаваемый пользователь»  
на языке с++**

**Курсовая работа**

Дисциплина: Высокоуровневые методы информатики и программирования

Студент: ИТ–30918 группы…………………………………. /Кандакова А.Н./

подпись

Направление:

Специальность:

Руководитель: к.п.н, доцент………………………………./Никифорова Т.А./

подпись

Комиссия: ….............................................................................../ /

….................................................................................../ /

….................................................................................../ /

Дата защиты:

Оценка:

ОГЛАВЛЕНИЕ

[ВВЕДЕНИЕ 3](#_Toc315364790)

[1 Разработка эскизного и технического проектов программы «Методы шифрования информации» 5](#_Toc315364791)

[1.1 Назначение и область применения приложения «Методы шифрования информации» 5](#_Toc315364792)

[1.2 Технические характеристики 5](#_Toc315364793)

[1.2.1 Постановка задачи 5](#_Toc315364794)

[1.2.2 Описание алгоритма 9](#_Toc315364796)

[1.2.3 Организация входных и выходных данных 10](#_Toc315364797)

[1.2.4 Выбор состава технических и программных средств 10](#_Toc315364798)

[1.3 Разработка внешнего вида приложения 11](#_Toc315364799)

[2 Разработка рабочего проекта ПРИЛОЖЕНИЯ «» 13](#_Toc315364800)

[2.1 Конструирование Windows-окна 13](#_Toc315364801)

[2.1.1 Визуальное проектирование приложения 13](#_Toc315364802)

[2.2 Написание кода программы 16](#_Toc315364803)

[2.3 Спецификация программы 24](#_Toc315364804)

[2.4 Описание программы и рекомендации по использованию 24](#_Toc315364805)

[2.5 Тестирование программы 26](#_Toc315364806)

[Заключение 27](#_Toc315364807)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 28](#_Toc315364808)

[ПРИЛОЖЕНИЕ 30](#_Toc315364809)

# ВВЕДЕНИЕ

# 1 Разработка эскизного и технического проектов программы «Методы шифрования информации»

## **1.1 Назначене и область применения приложения «Методы шифрования информации»**

Разрабатываемое приложение «Методы шифрования информации» предназначено для получения списка файлов и ключей реестра доступных для текущего пользователя OS Windows.

Разрабатываемое приложение может использоваться в плане информационной безопасности: ограничение доступа к определённым файлам для пользователей, мониторинг активности текущего пользователя, установления уровня доступа. Так же этот инструмент может быть полезен для расследований преступлений, совершаемые в сфере информационных технологий.

## **1.2 Технические характеристики**

### *1.2.1 Постановка задачи*

Целью курсовой работы является разработка приложения, которое будет предоставлять пользователю возможность шифрования/дешифрования файлов с использованием одного из предложенных алгоритмов: шифр Цезаря, Атбаш, шифр Вижинера. Такая программа должна быть проста в использовании, иметь удобный интерфейс, чтобы пользователь за несколько «кликов» мог зашифровать/дешифровать выбранный файл.

Далее следует краткое описание выбранных алгоритмов шифрования.

*Алгоритм Цезаря.*

Шифр Цезаря является частным случаем шифра простой замены (одноалфавитной подстановки). Свое название этот шифр получил по имени римского императора Гая Юлия Цезаря, который использовал этот шифр при переписке с Цицероном (около 50 г. до н.э.).

При шифровании исходного текста каждая буква заменялась на другую букву того же алфавита по следующему правилу. Заменяющая буква определялась путем смещения вправо по алфавиту от исходной буквы на К букв. При достижении конца алфавита выполнялся циклический переход к его началу. Цезарь использовал шифр замены при смещении К = 3. Такой шифр замены можно задать таблицей подстановок, содержащей соответствующие пары букв открытого текста и шифртекст. Совокупность возможных подстановок для К=3 показана в табл. 1.

*Таблица 1*

*Одноалфавитная подстановка (K=3, m=26)*

|  |  |  |
| --- | --- | --- |
| A→D | J→M | S→V |
| В→Е | K→N | T→W |
| C→F | L→0 | U→X |
| D→G | М→Р | V→Y |
| Е→Н | *N→Q* | *W→Z* |
| F→I | 0→R | X→A |
| G→J | P→S | Y→B |
| H→K | Q→T | Z→C |
| 1 I???→L | R→U |  |

Например, послание Цезаря «VENI V1DI VICI» (в переводе на русский означает "Пришел, Увидел, Победил"), направленное его другу Аминтию после победы над понтийским царем Фарнаком, сыном Митридата, выглядело бы в зашифрованном виде так: «YHQL YLGL YLFL».

Достоинством системы шифрования Цезаря является простота шифрования и расшифрования. К недостаткам системы Цезаря следует отнести следующие:

* подстановки, выполняемые в соответствии с системой Цезаря, не маскируют частот появления различных букв исходного открытого текста;
* сохраняется алфавитный порядок в последовательности заменяющих букв; при изменении значения К изменяются только начальные позиции такой последовательности;
* число возможных ключей К мало;
* шифр Цезаря легко вскрывается на основе анализа частот появления букв в шифртексте.

Криптоанализ системы одноалфавитной замены начинается с подсчета частот появления символов: определяется число появлений каждой буквы в шифртексте. Затем полученное распределение частот букв в шифртексте сравнивается с распределением частот букв в алфавите исходных сообщений, например в английском. Буква с наивысшей частотой появления в шифртексте заменяется на букву с наивысшей частотой появления в английском языке и т.д. Вероятность успешного вскрытия системы шифрования повышается с увеличением длины шифртекста.

Концепция, заложенная в систему шифрования Цезаря, оказалась весьма плодотворной, о чем свидетельствуют ее многочисленные модификации.

*Алгоритм Атбаш.*

Порой священные иудейские тексты шифровались методом замены. Вместо первой буквы алфавита писалась последняя буква, вместо второй - предпоследняя и т.д. Этот древний шифр назывался Атбаш.

Если попытаться изобразить механизм шифра Атбаш наглядно, то это может выглядеть следующим образом:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| А | Б | В | Г | Д | … | Э | Ю | Я | \_ |

### 

*Рисунок 1 - Механизм шифра Атбаш*

Так, например, для текста, состоящего только из букв кириллического алфавита и пробела, замена будет осуществляться по правилу: А-пробел, Б-Я, В-Ю и т.д. Абсолютно аналогично справедлива обратная замена.

*Шифр Вижинера.*

Система Вижинера впервые была опубликована в 1586 г. и является одной из старейших и наиболее известных многоалфавитных систем. Многоалфавитная система или шифр сложной замены представляет собой следующее: для шифрования каждого символа исходного сообщения применяют свой шифр простой замены. Многоалфавитная подстановка последовательно и циклически меняет используемые алфавиты. Свое название она получила по имени французского дипломата XVI века Блеза Вижинера, который развивал и совершенствовал криптографические системы. Система Вижинера подобна системе шифрования Цезаря, у которой ключ подстановки меняется от буквы к букве. Этот шифр многоалфавитной замены можно описать таблицей шифрования, называемой таблицей (квадратом) Вижинера. В таблице 2 приведена таблица Вижинера для русского алфавита.

*Таблица 2*

*Таблица Вижинера для русского алфавита*

| Ключ | Символы русского алфавита | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч | ш | щ | ь | ы | ъ | э | ю | я |
| 0 | а | б | в | г | д | е | ж | з | и | и | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч | ш | щ | ь | ы | ъ | э | ю | я |
| 1 | б | 3 | г | д | е | ж | з | и | и | к | л | м | н | о | п | р | с | т | у | 4 | х | ц | ч | ш | щ | ь | ы | ъ | э | ю | я | а |
| 2 | в | г | д | е | ж | з | и | и | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч | ш | щ | ь | ы | ъ | э | го | я | а | б |
| 3 | г | д | е | ж | з | и | и | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в |
| 4 | д | в | ж | з | и | и | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г |
| 5 | е | ж | з | и | и | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч | шu | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д |
| 6 | ж | з | и | и | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | е | г | д | е |
| 7 | з | и | и | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | 3 | г | д | е | ж |
| 8 | и | и | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з |
| 9 | и | к | л | м | н | о | п | р | с | т | у | ф | х | ч | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и |
| 10 | к | л | м | н | о | п | р | с | т | у | ф | х | ч | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | х | з | и | й |
| 11 | л | м | н | о | п | р | с | т | у | ф | х | ч | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к |
| 12 | м | н | о | п | р | с | т | у | ф | х | ч | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л |
| 13 | н | о | п | р | с | т | у | ф | х | ч | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л | м |
| 14 | о | п | р | с | т | у | ф | х | ч | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л | м | н |
| 15 | п | р | с | т | у | ф | х | ч | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о |
| 16 | р | с | т | у | ф | х | ч | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о | п |
| 17 | с | т | у | ф | х | ч | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о | п | р |
| 18 | т | у | ф | х | ч | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о | п | р | с |
| 19 | у | ф | х | ч | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о | п | р | с | т |
| 20 | ф | х | ц | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о | п | р | с | т | у |
| 21 | х | ц | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о | п | р | с | т | у | ф |
| 22 | ц | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о | п | р | с | т | у | ф | х |
| 23 | ч | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | а | а | ж | з | и | й | к | л | м | н | о | п | р | с | т | у | ф | х | ц |
| 24 | ш | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч |
| 25 | щ | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч | ш |
| 26 | ь | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч | ш | щ |
| 27 | ы | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч | ш | щ | ь |
| 28 | ъ | э | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч | ш | щ | ь | ы |
| 29 | э | ю | я | а | б | в | г | д | и | ж | з | и | й | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч | ш | щ | ь | ы | ъ |
| 30 | ю | я | а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч | ш | щ | ъ | ы | ъ | э |
| 31 | я | а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о | п | р | с | т | у | ф | х | ц | ч | ш | щ | ь | ы | ъ | э | ю |

Таблица Вижинера используется для зашифрования и расшифрования. Таблица имеет два входа:

• верхнюю строку подчеркнутых символов, используемую для считывания очередной буквы исходного открытого текста;

• крайний левый столбец ключа.

Последовательность ключей обычно получают из числовых значений букв ключевого слова.

При шифровании исходного сообщения его выписывают в строку, а под ним записывают ключевое слово (или фразу). Если ключ оказался короче сообщения, то его циклически повторяют. В процессе шифрования находят в верхней строке таблицы очередную букву исходного текста и в левом столбце очередное значение ключа. Очередная буква шифртекста находится на пересечении столбца, определяемого шифруемой буквой, и строки, определяемой числовым значением ключа.

### *1.2.2 Описание алгоритма*

При запуске приложения пользователю первоначально необходимо с методом шифрования. Далее, в зависимости от выбранного алгоритма шифрования, пользователь должен задать параметры шифра. Например, выбрав алгоритм шифрования Вижинера, пользователю следует задать величину сдвига и указать пароль. Эти данные будут необходимы для дешифрования.

Затем пользователю требуется указать файл для шифрования или дешифрования. Для этого предоставляется возможность либо указать путь к файлу, либо выбрать файл в открывшемся после нажатия кнопки «Открыть» окне. Далее необходимо выбрать действие над файлом: шифрование или дешифрование.

При возникновении сложностей в установлении параметров шифрования пользователь, нажав кнопку «Помощь», может получить краткий алгоритм работы с данным приложением.

### *1.2.3 Организация входных и выходных данных*

Входными данными для приложения является файл произвольной природы, предназначенный для шифрования. Алгоритм шифрования применяется к битовой структуре файла. По умолчанию программа добавляет префикс “enc\_” к имени зашифрованного файла и сохраняет его в исходный каталог.

Выходными данными будет зашифрованный ранее файл произвольной природы с префиксом “enc\_” в имени, предназначенный для дешифрования. Алгоритм шифрования применяется к битовой структуре файла. По умолчанию программа добавляет префикс “dec\_” к имени зашифрованного файла и сохраняет его в исходный каталог.

### *1.2.4 Выбор состава технических и программных средств*

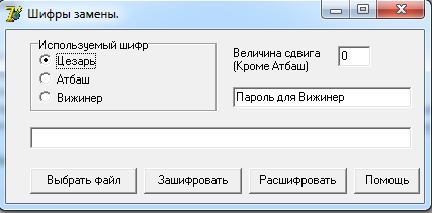
Средой разработки приложения была выбрана интегрированная среда разработки ПО для Microsoft Windows – Borland Delphi 7. Ядром данной среды программирования является объектно-ориентированный язык программирования Object Pascal.

Выбор данной среды разработки обусловлен следующими причинами:

1. Delphi – объектно-ориентированная среда программирования.
2. Визуальное (а, следовательно, и скоростное) построение приложений из программных прототипов.
3. Наличие большого числа визуальных и невизуальных предустановленных компонент (например, стандартный диалог для открытия файла и др.)
4. Полная поддержка Unicode. Приложения могут выполняться на любой языковой версии Windows. Применение Unicode гарантирует, что приложения будут одинаково выглядеть и функционировать во всех языковых версиях Windows и поддерживать как Unicode-строки, так и ANSI-строки.

## **1.3 Разработка внешнего вида приложения**

Интерфейс приложения должен быть наиболее удобным и простым в использовании, поэтому на форме приложения компоненты будут располагаться компактно.

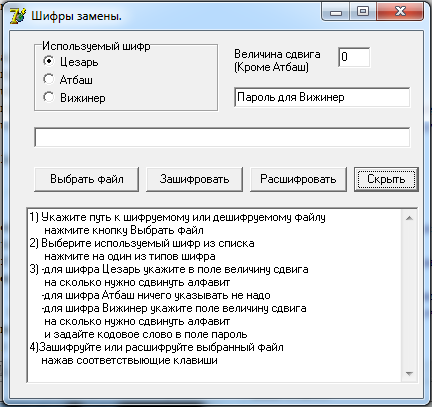


*Рисунок 2 - Внешний вид работающего приложения*

Для удобства пользователя при выборе файла для шифрования форма имеет кнопку «Выбрать файл», которая запускает стандартный диалог для открытия файла; кнопки «Расшифровать», «Зашифровать» для выполнения выбранного алгоритма шифрования с заданными параметрами.

Также на форме имеются поля ввода для указания параметров шифрования или дешифрования.

При возникновении у пользователя затруднений предусмотрена кнопка «Помощь», по нажатию на которую будет отображено описание краткого алгоритма работы с приложением.

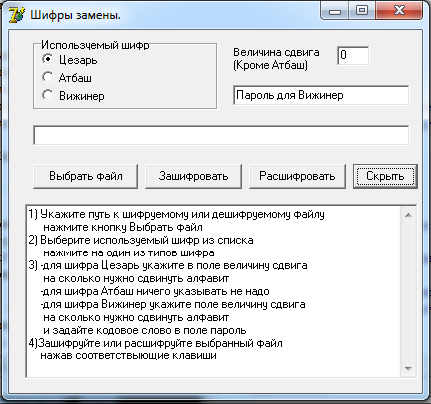


*Рисунок 3 - Внешний вид работающего приложения при нажатии кнопки «Помощь»*

# 2 Разработка рабочего проекта ПРИЛОЖЕНИЯ «мЕТОДЫ ШИФРОВАНИЯ ИНФОРМАЦИИ»

## **2.1 Конструирование Windows-окна**

### 2.1.1 Визуальное проектирование приложения

****

***TMemo***

***TEdit***

***TEdit***

***TEdit***

***TRadioGroup***

***TForm***

***TButton***

***TButton***

***TButton***

***TButton***

*Рисунок 4 - Визуальные и невизуальные компоненты главной формы приложения*

*Таблица 3*

*Назначение компонентов главной формы приложения*

| ***№ п/п*** | ***Компонент*** | ***Назначение*** |
| --- | --- | --- |
| 1 | Main | Главная форма приложения |
| 2 | TRadioGroup1 | Выбора пользователем одной из нескольких взаимоисключающих альтернатив |
| 3 | ButOpen | Кнопка запуска стандартного диалога открытия файла |
| 4 | ButEncrypt | Кнопка запуска шифрования файла |
| 5 | ButDecrypt | Кнопка запуска дешифрования файла |
| 6 | ButHelp | Кнопка вызова и скрытия справки |
| 7 | Label1 | Надпись |
| 8 | Memo1 | Поле для вывода текста справки |
| 9 | EdShift | Поле для ввода значения сдвига |
| 10 | EdPass | Поле для ввода пароля |
| 11 | EdFileName | Поле для отображения имени исходного файла |
| 12 | FileOpenDlg | Запуск стандартного диалога открытия файла |

*Таблица 4*

*Изменённые свойства компонентов главной формы приложения*

| *№ п/п* | *Компонент* | *Свойства* |
| --- | --- | --- |
| 1 | MainFrm | ClientHeight = 170  ClientWidth = 413  OnCloseQuery = FormCloseQuery  OnResize = FormResize  Color = clActiveCaption  Caption = ‘Шифры замены‘  HorzScrollBar = (TControlScrollBar)  OldCreateOrder = False  Width = 429 |
| 2 | TRadioGroup1 | Height = 73  Width = 185  Left = 24  Top = 8  Caption = ‘Используемый шифр’  Hint = ‘Выберите шифр’  ParentShowHint = False  ShowHint = True  TabOrder = 0 |
| 3 | Label1 | Caption = ‘Величина сдвига (Кроме Атбаш)’  Height = 26  Width = 86  Hint = ‘поле величины сдвига находится правее’  Left = 224  ParentShowHint = False  ShowHint = True  Top = 16  WordWrap = True |
| 4 | EdShift | Height = 21  Width = 33  Top = 16  Hint = ‘Сдвиг Алфавита’  Left = 328  ParentShowHint = False  TabOrder = 1  ShowHint = True |
| 5 | EdPass | Height = 21  Cursor = crDrag  Hint = ‘Введите пароль для алгоритма Вижинера’  Left = 224  Top =56  Width = 177  TabOrder = 2  ParentShowHint = False  ShowHint = True |
| 6 | EdFileName | Height = 21  Hint = ‘Введите пароль для алгоритма Вижинера’  Left = 24  Top =96  Width = 377  TabOrder = 3  ReadOnly = True |
| 7 | FileOpenDlg | Filter = ‘Текстовый файл|\*.txt|Любой файл|\*.\*’  InitialDir = C |
| 8 | ButOpen | Caption = ‘Выбрать файл’  Height = 25  Left = 24  Hint = ‘Нажмите для выбора шифруемого файла’  ParentShowHint = False  ShowHint = True  TabOrder = 4  Top = 136  Width = 105  OnClick = ButOpenClick |
| 9 | ButEncrypt | Caption = ‘Зашифровать’  Height = 25  Left = 136  Hint = ‘Зашифровать выбранный файл’  ParentShowHint = False  ShowHint = True  TabOrder = 5  Top = 136  Width = 97  OnClick = ButEncryptClick |
| 10 | ButDecrypt | Caption = ‘Расшифровать’  Height = 25  Left = 240  Hint = ‘Расшифровать выбранный файл’  ParentShowHint = False  ShowHint = True  TabOrder = 6  Top = 136  Width = 97  OnClick = ButDecryptClick |
| 11 | ButHelp | Caption = ‘Помощь’  Height = 25  Left = 344  TabOrder = 8  Top = 136  Width = 65  OnClick = ButHelpClick |
| 12 | Memo1 | Height = 177  Left = 16  Width = 188  ReadOnly = True  ScrollBars = ssVertical  TabOrder = 7  Top = 176  Width = 393  Visible = False |

## **2.2 Написание кода программы**

При работе программы использованы следующие процедуры:

*Таблица 5*

*Используемые процедуры и функции в главной форме приложения*

| *Процедура* | *Назначение* |
| --- | --- |
| TMainFrm.ButOpenClick(Sender: TObject); | Выбор файла для шифрования или дешифрования |
| TMainFrm.ButEncryptClick(Sender: TObject); | Проверка наличия исходного файла для шифрования |
| TMainFrm.ButDecryptClick(Sender: TObject); | Проверка наличия исходного файла для дешифрования |
| TMainFrm.FileEncrypt(); | Процедура шифрования файла |
| TMainFrm.FileDecrypt(); | Процедура дешифрования файла |
| TMainFrm.Enc\_Ceaser(iCount: Integer); | Процедура шифрования по алгоритму Цезаря |
| TMainFrm.Dec\_Ceaser(iCount: Integer); | Процедура дешифрования по алгоритму Цезаря |
| TMainFrm.Enc\_Atbash(iCount: Integer); | Процедура шифрования по алгоритму Атбаш |
| TMainFrm.Dec\_Atbash(iCount: Integer); | Процедура дешифрования по алгоритму Атбаш |
| TMainFrm.Enc\_Vision(iCount: Integer); | Процедура шифрования по алгоритму Вижинера |
| TMainFrm.Dec\_Vision(iCount: Integer); | Процедура дешифрования по алгоритму Вижинера |
| TMainFrm.ButHelpClick(Sender: TObject); | Вызов справки |

Приведём описание некоторых из вышеописанных процедур главной формы приложения «Методы шифрования информации».

Список процедур главной формы приложения для более подробного описания:

процедуры, возникающие при нажатии на кнопки:

procedure ButOpenClick(Sender: TObject);

procedure ButEncryptClick(Sender: TObject);

procedure ButDecryptClick(Sender: TObject);

procedure FileEncrypt(); //процедура зашифровки файла

procedure FileDecrypt(); //процедура расшифровки файла

кодирующие и декодирующие процедуры по выбранным алгоритмам:

procedure Enc\_Ceaser(iCount: Integer);

procedure Enc\_Atbash(iCount: Integer);

procedure Enc\_Vision(iCount: Integer);

procedure Dec\_Ceaser(iCount: Integer);

procedure Dec\_Atbash(iCount: Integer);

procedure Dec\_Vision(iCount: Integer);

procedure ButHelpClick(Sender: TObject); - процедура вызова справки

Сначала определяем размер буфера загрузки и выгрузки данных:

var

MainFrm: TMainFrm;

InpFileName : AnsiString; //входящий буфер

OutFileName : AnsiString; //записывающий буфер

inp\_buffer : array [0..511] of byte; //определяем размеры буфера

out\_buffer : array [0..511] of byte; //загрузки и выгрузки данных

Затем выбираем исходный файл для шифрования или дешифрования:

procedure TMainFrm.ButOpenClick(Sender: TObject); {выбираем нужный файл}

begin

if (FileOpenDlg.Execute) then //если файл открыт, то..

begin

InpFileName := FileOpenDlg.FileName; //запоминаем нужный файл

EdFileName.Text := InpFileName; //и сохраняем путь

end;

end;

Далее осуществляем проверку наличия файла для шифрования или дешифрования:

procedure TMainFrm.ButEncryptClick(Sender: TObject); {проверка наличия файла}

begin

if FileExists(InpFileName) //если есть шифруемый файл,

then FileEncrypt() //то шифруем его,

else ShowMessage('Не указан шифруемый файл!'); {иначе вывести сообщение}

end;

procedure TMainFrm.ButDecryptClick(Sender: TObject); {проверка наличия файла}

begin

if FileExists(InpFileName) //если есть раcшифруемый файл,

then FileDecrypt() //то раcшифруем его,

else ShowMessage('Не указан расшифровываемый файл!'); {иначе вывести сообщение}

end;

Подробно опишем центральные процедуры приложения – процедуры шифрования и дешифрования файла.

procedure TMainFrm.FileEncrypt(); //Процедура зашифровки файла

var

fsInp : TFileStream; //входная переменная

fsOut : TFileStream; //выходная переменная

iCnt : Integer; //счетчик проверка

begin

OutFileName := ExtractFilePath(InpFileName) + 'Enc\_' + ExtractFileName(InpFileName); {Имя выходного файла := путь к файлу + 'Enc\_' + имя шифруемого файла}

fsInp := TFileStream.Create(InpFileName, fmOpenRead); {входную переменную мы определяем как для чтения}

fsOut := TFileStream.Create(OutFileName, fmCreate); {выходную переменную мы создаем}

fsInp.Seek(0, soFromBeginning); //переходим на начало

iCnt := -1; //присваиваем значение не равное нулю

while (iCnt <> 0) do {до тех пор пока iCnt <> работаем с буфером данных}

begin

iCnt := fsInp.Read(inp\_buffer, 512); {Читаем последний символ блока}

{в зависимости от выбранного пункта в RadioGrp используем соответствующий тип шифрования}

case RadioGrp.ItemIndex of

0: Enc\_Ceaser(iCnt);

1: Enc\_Atbash(iCnt);

2: Enc\_Vision(iCnt);

end;

fsOut.Write(out\_buffer, iCnt); //выгружаем полученные данные

end;

fsOut.Free;

fsInp.Free; //освобождаем переменные

end;

procedure TMainFrm.FileDecrypt(); //Процедура дешифрования файла

var

fsInp : TFileStream; //входная переменная

fsOut : TFileStream; //выходная переменная

iCnt : Integer; //счетчик проверка

begin

OutFileName := ExtractFilePath(InpFileName) + 'Dec\_' + ExtractFileName(InpFileName); {Имя выходного файла := путь к файлу + 'Dec\_' + имя дешифруемого файла}

fsInp := TFileStream.Create(InpFileName, fmOpenRead); {входную переменную мы определяем как для чтения}

fsOut := TFileStream.Create(OutFileName, fmCreate); {выходную переменную мы создаем}

fsInp.Seek(0, soFromBeginning); //переходим на начало

iCnt := -1; //присваиваем значение не равное нулю

while (iCnt <> 0) do {до тех пор пока iCnt не равен нулю работаем с буфером данных}

begin

//Читаем последний символ блока

iCnt := fsInp.Read(inp\_buffer, 512);

{в зависимости от выбранного пункта в RadioGrp используем свой тип дешифрования}

case RadioGrp.ItemIndex of

0: Dec\_Ceaser(iCnt);

1: Dec\_Atbash(iCnt);

2: Dec\_Vision(iCnt);

end;

fsOut.Write(out\_buffer, iCnt); //выгружаем полученные данные

end;

fsOut.Free;

fsInp.Free; //освобождаем переменные

end;

Описание процедур, осуществляющих шифрование или дешифрование в зависимости от выбранного пользователем алгоритма.

procedure TMainFrm.Enc\_Ceaser(iCount: Integer); {шифрование по алгоритму Цезаря}

var

i : Integer; //переменная для цикла

Shift : byte; //величина сдвига

begin

Shift := byte(StrToInt(EdShift.Text)); {величина на которую шифрует римский круг}

for i := 0 to iCount do // цикл обработки данных

begin

out\_buffer[i] := (inp\_buffer[i] + Shift) mod 256; {выходной символ:=(входной символ + сдвиг) по модулю 256;}

end;

end;

procedure TMainFrm.Dec\_Ceaser(iCount: Integer); {дешифрование по алгоритму Цезаря}

var

i : Integer; //переменная для цикла

Shift : byte; //величина сдвига

begin

Shift := byte(StrToInt(EdShift.Text)); {величина на которую дешифрует римский круг}

for i := 0 to iCount do // цикл обработки данных

begin

out\_buffer[i] := (inp\_buffer[i] + 256 - Shift) mod 256; {выходной символ:=(входной символ + 256 - сдвиг) по модулю 256;}

end;

end;

procedure TMainFrm.Enc\_Atbash(iCount: Integer); {шифрование по алгоритму Атбаш}

var

i : Integer; //переменная для цикла

begin

for i := 0 to iCount do // цикл обработки данных

begin

out\_buffer[i] := 256 - inp\_buffer[i]; {выходной символ:=256 - входной символ;}

end;

end;

procedure TMainFrm.Dec\_Atbash(iCount: Integer); {дешифрование по алгоритму Атбаш}

var

i : Integer; //переменная для цикла

begin

for i := 0 to iCount do // цикл обработки данных

begin

out\_buffer[i] := 256 - inp\_buffer[i]; {выходной символ:=256 - входной символ;}

end;

end;

procedure TMainFrm.Enc\_Vision(iCount: Integer); {дешифрование по алгоритму Вижинера}

var

i, s : Integer;//переменные для циклов

Shift: byte; //величина сдвига

Cod : string; //кодовое слово

begin

Shift := byte(StrToInt(EdShift.Text)); {величина на которую на которую смещается шифр}

Cod:= EdPass.Text; //определяем кодовое слово

s:=1; //определяем с какого символа будем использовать кодовое слово

for i := 0 to iCount do // цикл обработки данных

begin

out\_buffer[i] := (inp\_buffer[i] + byte(Cod[s]) + Shift) mod 256; {выходной символ:=(входной символ + символ кодового слова + сдвиг) по модулю 256;}

Inc(s); //

if (s = length(Cod) + 1) then s:= 1; {блок который не позволяет выйти за кодовое слово}

end;

end;

procedure TMainFrm.Dec\_Vision(iCount: Integer); {дешифрование по алгоритму Вижинера}

var

i, s : Integer;//переменные для циклов

Shift: byte; //величина сдвига

Cod : string; //кодовое слово

begin

Shift := byte(StrToInt(EdShift.Text)); {величина, на которую смещается шифр}

Cod:= EdPass.Text; //определяем кодовое слово

s:=1; //определяем с какого символа будем использовать кодовое слово

for i := 0 to iCount do // цикл обработки данных

begin

out\_buffer[i] := (inp\_buffer[i] + 256 - byte(Cod[s]) - Shift) mod 256; {выходной символ:=(входной символ + 256 - символ кодового слова - сдвиг) по модулю 256;}

Inc(s);

if (s = length(Cod) + 1) then s:= 1; {блок который не позволяет выйти за кодовое слово}

end;

end;

## **2.3 Спецификация программы**

Состав проекта

|  |  |
| --- | --- |
| *Наименование* | *Обозначение* |
| Методы шифрования.exe | Исполняемый файл приложения (Windows-приложение) |

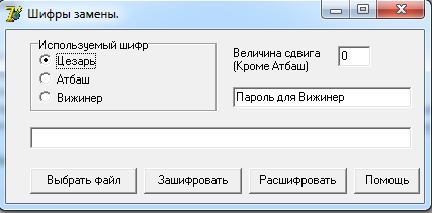
## **2.4 Описание программы и рекомендации по использованию приложения**

Данная программа проста в использовании. Для запуска достаточно запустить файл «Методы шифрования.exe» на компьютере.

Для работы с данной программой ненужно знать все особенности описанных выше методов шифрования. Любой, даже начинающий пользователь сможет применять приложение без особых трудностей.

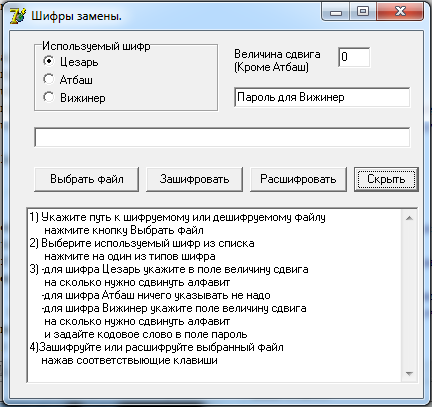
Программа пригодна для демонстрации простых алгоритмов шифрования, также выполняет свою основную функцию – шифрование или дешифрование файлов.

После запуска файла «Методы шифрования.exe» откроется главное окно приложения, в котором следует выбрать алгоритм шифрования, указать файл для шифрования, задать величину сдвига (только при использовании алгоритмов Цезаря и Вижинера) и пароль (только при использовании шифра Вижинера) и указать направление (шифровать или дешифровать).



*Рисунок 5 - Внешний вид работающего приложения*

При возникновении у пользователя затруднений можно отобразить краткое описание алгоритма работы с приложением.



*Рисунок 6 - Внешний вид работающего приложения при нажатии кнопки «Помощь»*

## **2.5 Тестирование программы**

Программа успешно прошла тестирование, не найдено ошибок при шифровании, дешифровании файлов произвольной природы. Приложение шифрует, дешифрует файлы по выбранному алгоритму, осуществляет работу с входными и выходными данными.

# **Заключение**

# **СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

# **ПРИЛОЖЕНИЕ**

Структура диска: