САНКТ-­ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра КТИ

КУРСОВОЙ ПРОЕКТ

по дисциплине «Программирование»

Тема: Стеганография в изображении

Выполнил:   Иванов А.И. Группа:  6361

Факультет:  КТИ

Проверил:  Пелевин М.С.

Оценка:

Дата: «  » декабря 2016 г.

Санкт-­Петербург

2016

Оглавление

1. Введение3
2. Описание предметной области4
3. Кодирование данных в изображении форматов без сжатия5
4. Практическая реализация7
5. Работа с файлами9
6. Шифрование/расшифровывание 10
7. Работа с программой11
8. Заключение13
9. **Введение**

Целью курсового проекта является написание программы, в которой реализовано шифрование файла (в частности текстового документа) в графический файл формата \*.BMP . Исходный файл шифруется в изображение с помощью стеганографического метода, прячущего его в цветовые каналы RGB. Результатом курсового проекта стало получение консольной программы с возможностями шифрования и дешифрования документа в изображение. В отчете изложена информация об основных стеганографических методах шифрования и готовом функционале написанной программы.

1. **Описание предметной проблемы**

Стеганографическая система (стегосистема) — объединение методов и средств используемых для создания скрытого канала для передачи информации. Стеганография в наше время часто применяется, как правило, для встраивания цифровых водяных знаков, являющееся основой для систем защиты авторских прав и DRM (Digital rights management) систем.

Также в качестве данных, в которые передаются исходные,  могут быть использованы изображения, текстовые массивы,  аудиофайлы. Именно это и будет реализовано в данной работе.

1. **Кодирование данных в изображения форматов без сжатия**

Шифрование файла в изображении реализовано с помощью метода LSB (Least Significant Bit, наименьший значащий бит). Его суть заключается в замене последних значащих битов в контейнере (изображения, аудио или видеозаписи) на биты скрываемого файла. Для эффективного выполнения задачи разница между пустым и заполненным контейнерами должна быть минимальна (не ощутима для органов восприятия человека).

Исходные данные (cам файл и его размер) будут “спрятаны” в изображении формата\*.BMP

*BMP* для представления цвета использует цветовую модель *RGB*, т.е. цвет, который видит человек получается в результате смешивания трех цветов Red, Green, Blue (красного, зелёного, синего), также *BMP* обычно не использует сжатие, что дает возможность спрятать в нем достаточно большое количество информации.

Так, один байт файла **10 101 010** будет расположен в пикселе изображения следующим образом:

R: 111100**10**  
G: 00001**101**  
B: 11001**010**

Также для более защищенной передачи данных к исходному текстовому файлу будет применем шифр Цезаря— один из самых простых и наиболее широко известных методов шифрования.

Расшифровывание будет проходить по следующему принципу:

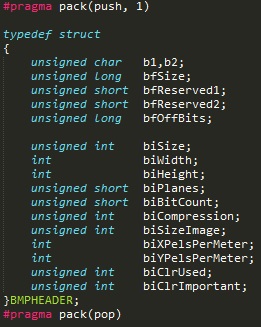
* Из каналов R,G,B считываются первые два байта “спрятанных” данных (по схеме 2-3-3), в которых указан размер зашифрованного файла.
* Последующие значения малозначащих битов записываются до тех пор , пока размер полученного файла не достигнет первоначального значения .
* Если зашифрован был текстовый файл ,то к нему будет применен обратный шифр Цезаря.

1. **Практическая реализация**

Ниже приведена таблица, показывающая назначение полей структуры BMP файла.

****

Для размещения в памяти заголовка файла  и и информации об изображении в программе используется следующая структура:



Размер этой структуры в памяти зависит от настроек компилятора и от директив в коде.  
  
Поля выравниваются в памяти по границе кратной своему же размеру(1-байтовые поля не выравниваются , 2-байтовые — выравниваются на чётные позиции, 4-байтовые — на позиции кратные четырём и т.д. В большинстве случаев выравнивание размера структуры в памяти составляет 4 байта.

Если в случае обычной структуры все поля внутри неё выравниваются на выравнивание базового типа поля, то при наличии *#pragma pack* это выравнивание ограничивается. В случае единицы все поля структуры будут идти впритык друг к другу без выравниваний , что позволит избежать ошибок в работе программы.

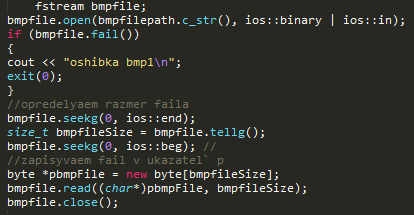
1. **Работа с файлами**

В программе присутствуют функции записи и считывания из картинки (*pack* и *unpack* соответственно) .

Для работы с файлами используется класс fstream. В этом классе реализованы функции открытия и закрытия файла, а также члены класса, которые позволяют позиционировать указатель**.**

****

Открываем файл формата \*.BMP , определяем его размер , записываем в указатель.

****

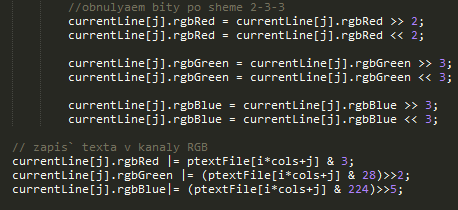
Открываем файл для шифрования, определяем его размер, записываем в указатель.

Аналогичная процедура в функции *unpack .*

1. **Шифрование/расшифровывание**

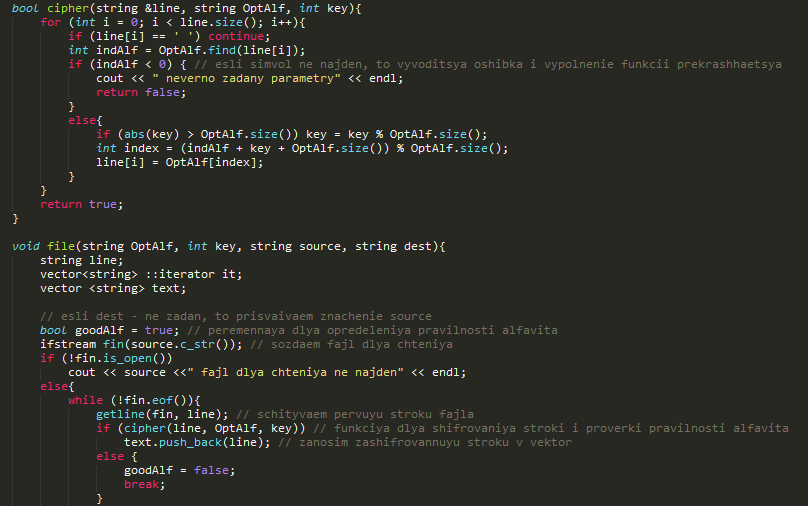
Процесс записи в каналы R,G,B пикселя реализованы в

следующих строках:

****

Считывание :

****

Также реализован шифр Цезаря в функциях cipher и file

Сам проект доступен для скачивания по ссылке:

**https://github.com/MaminDrug/stega**

1. **Работа с программой**

Для работы программы нужно ввести :

$ stega [textfile] <OptType> [<bmpfile>] <key>

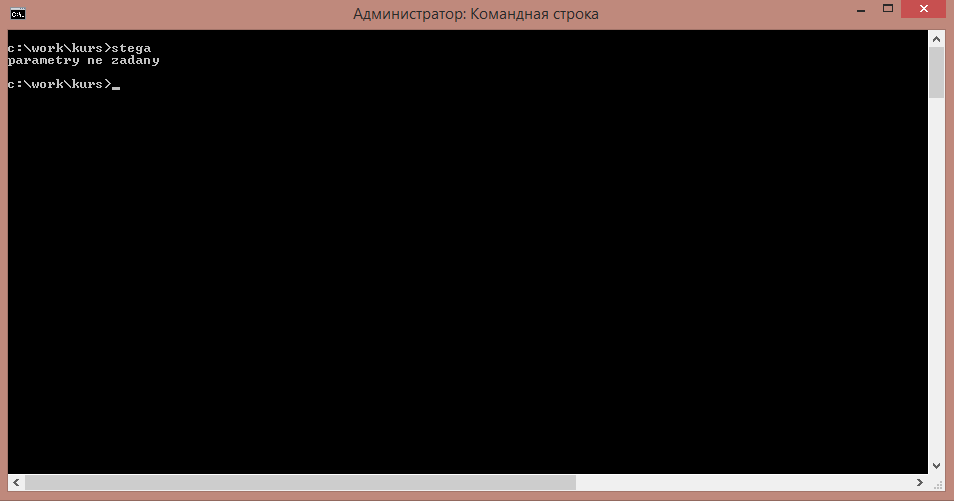
OptType : type может быть 'encode' или 'decode'.

Key : ключ для шифрования/дешифрования.

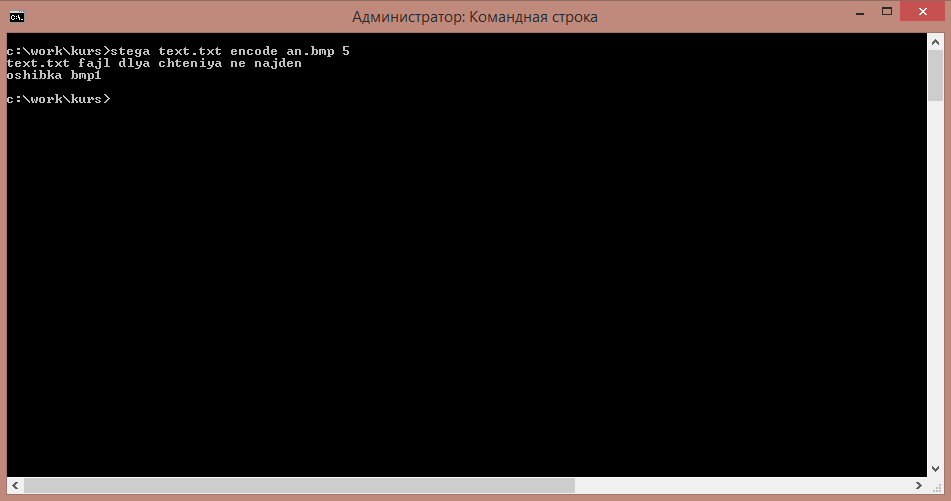
textfile : путь к txt файлу.

bmpfile : путь к bmp файлу.

Некорректно введенные параметры приведут к ошибке :



Указание пути на несуществующий файл также приведет к ошибке :



1. **Заключение**

Программа позволяет ” прятать “ файл в изображение формата \*.*BMP* с помощью метода LSB.

**Преимущества метода:**

1)     простота использования;

2)     простота реализации;

3)     возможность скрытой передачи большого объема информации;

**Недостатки метода:**

1. низкая защищенность;

По чересчур большому размеру картинки легко можно заподозрить в шифровании. Для того, чтобы улучшить защищенность передачи информации можно использовать шифр Цезаря либо же записывать в пиксели изображения , которые будут выбраны по заданному ключу.