МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

кафедра Компьютерных технологий

«Допущен к защите»
Заведующий кафедрой д.ф.-м.н.
профессор Курмбаев З.К.
(Ф.И.О., ученая степень, звание)
_____ «____» _____ **20__ г.**
(подпись)

ДИПЛОМНЫЙ ПРОЕКТ

На тему: Разработка программного продукта для стеганографического скрытия информации

Специальность 050704 – „Вычислительная техника и програм.обесп."

Выполнил (а) Нургазинов А.Е. ВТ-10-3
(Фамилия и инициалы) группа

Научный руководитель Шайхин Б.М. к.ф.-м.н., доц. КТ
(Фамилия и инициалы, ученая степень, звание)

Консультанты:

по экономической части:
Ержанова З.Ю., ст. преп.
(Фамилия и инициалы, ученая степень, звание)
Ержанова «14» 05 2014 г.
(подпись)

по безопасности жизнедеятельности:
Дуйсенова Н.Б., д.т.н. профессор
(Фамилия и инициалы, ученая степень, звание)
_____ «14» 05 2014 г.
(подпись)

по применению вычислительной техники:
Шайхин Б.М. к.ф.-м.н., доц. КТ
(Фамилия и инициалы, ученая степень, звание)
_____ «29» 05 2014 г.
(подпись)

Нормоконтролер: Тусупов Д.М.
(Фамилия и инициалы, ученая степень, звание)
_____ «30» мая 2014 г.
(подпись)

Рецензент: Мухамбаев М.К. д.т.н., доцент
(Фамилия и инициалы, ученая степень, звание)
_____ «31» мая 2014 г.
(подпись)

Алматы 2014 г.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

Факультет Информационных технологий
Специальность Вычислительная техника и программное обеспечение
Кафедра Компьютерных технологий

ЗАДАНИЕ
на выполнение дипломного проекта

Студент Нургазинов Айтуар Ерланович
(фамилия, имя, отчество)

Тема проекта Разработка программного продукта для стеганографического скрытия информации

утверждена приказом ректора № 115 от «24» сентября 2013 г.
Срок сдачи законченной работы «___» июня 2014 г.
Исходные данные к проекту требуемые параметры результатов проектирования (исследования) и исходные данные объекта

Разработать программный продукт для стеганографического скрытия информации в графических файлах. Топологию программного продукта следует разработать с учетом возможности передачи и принятия файлов по интернету, а также по локальной сети, при этом она должна обеспечивать должный уровень надежности и безопасности.

Перечень подлежащих разработке дипломного проекта вопросов или краткое содержание дипломного проекта:

1. Понятие, определение стеганографии, а также протоколы стеганографических систем.
2. Анализ методов встраивания информации в пространственную область изображения.
3. Разработка программного продукта
4. Расчет экономической целесообразности проекта
5. Анализ условий труда и безопасности жизнедеятельности

Перечень графического материала (с точным указанием обязательных чертежей)

Структурная схема стеганосистемы как системы связи
Соотношение методов стеганографии и стеганоанализа
Основная форма программы
Окно загрузки изображения

Рекомендуемая основная литература

1. Артемин Б.В., Стеганография. Журнал „Защита информации. Конфидент", 1996 – №4 – С. 47-50
2. Шведченко И.В., Анализ криптостеганографических алгоритмов. Проблемы управления и информатики, 2007 – №4 – С. 149-155
3. Генне О.В., Основные положения стеганографии. Журнал „Защита информации. Конфидент", 2000 -13.
4. Грибунин В.Г., Оков И.Н., Туринцев И.В. Цифровая стеганография 2002 – 272 с.

Консультанты по проекту с указанием относящихся к ним разделов

Раздел	Консультант	Сроки	Подпись
Экономика	Брежнева З.О.	09.04-14.05.14	Брежнева
БЖД	Фирсов К.Р.	11.04-14.05.14	
Нормоконтроль	Тусупов Д.М.	30.05.14г.	

Г Р А Ф И К
подготовки дипломного проекта

№ п/п	Наименование разделов, перечень разрабатываемых вопросов	Сроки представления руководителю	Примечание
1	Понятие, определение стеганографии, а также протоколы систем	3.03.2014	
2	Анализ методов встраивание изображение в информацию.	3.04.2014	
3.	Разработка программного продукта	3.05.2014	
4.	БЖД	14.05.2014	
5.	Экономика	14.05.2014	

Дата выдачи задания «3» марта 2014 г.

Заведующий кафедрой _____ _____
(подпись) (Фамилия и инициалы)

Руководитель _____ Шайхин Б.Ж.
(подпись) (Фамилия и инициалы)

Задание принял к исполнению
студент _____ Нургазинов А.Е.
(подпись) (Фамилия и инициалы)

АҢДАТПА

Осы берілген дипломдық жобада мәліметтің стеганографиялық жасыруына арналған программалық өнімнің жасалуы қарыстырылған. Программалық өнімнің оңделуі стеганографияның алгоритмін зерттеу мен сипаттау кезеңдерінен тұрады.

Мәліметті суреттеменің кеңістік шегіне енгізілу тәсілдерінің талдауы өткізілген. Жобада оған қоса өнімнің тиімділік көрсеткіштері әрі өтелімділік мерзімдерімен бизнес-үлгісі көрсетілген. Және де еңбек қоргау мен қауіпсіздік шаралары қарыстырылған.

АННОТАЦИЯ

В данном дипломном проекте рассматривается разработка программного продукта для стеганографического скрытия информации. Разработка программного продукта включает в себя этапы исследования, описания алгоритмов стеганографии.

Проведен анализ методов встраивания информации в пространственную область изображения. В проекте прилагается бизнес-модель продукта с показателями эффективности и сроков окупаемости. Так же рассмотрены меры техники безопасности и охраны труда.

ANNOTATION

In this diploma draft consider laboration of programm product fot steganography to hide information. Elaboration of programm product includes stage of research, descriptions algoritms description on of steganography.

The analysis of methods of embedding information in a spafal region of the image. The project included the business model of the product with the indicators of effcluency and paycack period. Also discussed measures of safety and labour production.

Введение

Надежная защита информации от постороннего доступа является актуальной задачей, но не решение проблемы в целом. Одно из перспективных направлений защиты информации сложили новые методы стеганографии. Слово *стеганография* в переводе с греческого дословно означает *тайнопись* (steganos — тайна, секрет; graphy — запись).

Стеганография представляет собой множество методов, основывающихся на различных правилах, которые обеспечивают скрытие самого факта нахождения тайной информации в той или иной среде, а также средств реализации этих методов. К ней можно отнести огромное множество секретных средств связи, таких как невидимые чернила, микрофотоснимки, условное расположение знаков, тайные (скрытые) каналы, средства связи с плавающими частотами, голография и т.д.

В настоящее время развиваются методы *компьютерной стеганографии* — самостоятельного научного направления информационной безопасности, изучающей проблемы создания компонентов скрываемой информации в открытой информационной среде, которая может быть сформирована вычислительными системами и сетями. Особенностью стеганографического подхода является то, что он не предусматривает прямого оглашения факта существования защищаемой информации. Это обстоятельство позволяет в рамках традиционно существующих информационных потоков или информационной среды решать некоторые важные задачи защиты информации ряда прикладных областей [1].

Основным определяющим моментом в стеганографии является стеганографическое преобразование. До недавнего времени стеганография, как наука, в основном изучала отдельные методы сокрытия информации и способы их технической реализации. Разнообразие принципов, заложенных в стеганографических методах, по существу тормозило развитие стеганографии как отдельной научной дисциплины и не позволило ей сформироваться в виде некоторой науки со своими теоретическими положениями и единой концептуальной системой, которая обеспечила бы формальное получение качественных и количественных оценок стеганометодов. В этом история развития стеганографии резко отличается от развития криптографии.

До конца XIX века стеганография и криптография развивались в рамках единой науки о тайнописи. После формулирования голландским офицером Кирхгоффсом (A. Kerckhoffs) знаменитого правила о том, что стойкость криптографического алгоритма должна определяется исключительно стойкостью ключа, криптография как отдельная наука отделилась от стеганографии. За последние десятилетия криптология из совокупности специальных методов превратилась в наукоемкую дисциплину, основанную на фундаментальных исследованиях из теории вероятности, математической статистики, чисел, алгебраических полей, что позволило ей решить ряд важных для практического применения задач. Например, определение стойкости зашифрованных сообщений по отношению к

возможным средствам криптоанализа, а также целый ряд других задач, решение которых позволяет получать достаточно четкие количественные характеристики средств криптографической защиты информации [2].

Наблюдаемый в настоящее время интерес к стеганографии, как совокупности методов сокрытия информации, возник в большой мере благодаря интенсивному внедрению и широкому распространению средств вычислительной техники во все сферы деятельности человека. В рамках вычислительных сетей возникли достаточно широкие возможности по оперативному обмену различной информацией в виде текстов, программ, звука, изображений между любыми участниками сетевых сеансов независимо от их территориального размещения. Это позволяет активно применять все преимущества, которые дают стеганографические методы защиты.

Стеганографические методы находят все большее применение в оборонной и коммерческой сферах деятельности в силу их легкой адаптируемости при решении задач защиты информации, а также отсутствия явно выраженных признаков средств защиты, использование которых может быть ограничено или запрещено (как, например, криптографических средств защиты).

Сегодня стеганографические технологии активно используются для решения следующих основных задач:

1) защиты информации с ограниченным доступом от несанкционированного доступа;

2) защиты авторских прав на некоторые виды интеллектуальной собственности;

3) преодоления систем мониторинга и управления сетевыми ресурсами;

4) камуфляжа программного обеспечения;

5) создания скрытых каналов утечки чувствительной информации от законного пользователя.

Применение стеганографий является одним из эффективных методов при решении задач *защиты информации с ограниченным доступом*. К примеру, одна секунда оцифрованного звука с частотой 44100 Гц и уровнем отсчета 8 бит в стерео режиме позволяет скрыть за счет замены младших битов на скрываемое сообщение около 10 Кбайт информации. Данное изменение не обнаруживается при проигрывании файла, т.к. значение отсчетов составляет менее 1%.

3 Разработка программного продукта

На основе метода замены наименее значащего бита был разработан программный продукт на языке C++ Builder 2009.

Программный продукт может работать в двух режимах:

1) Шифрование — используется для встраивания информации в изображение;

2) Дешифрование — используется для извлечения встроенной в изображение информации.

Листинг модулей основной формы программного продукта приведен в Приложение А.

Листинг открытия ключа шифрования приведен в Приложении Б.

Листинг сохранения ключа шифрования приведен в Приложении В.

В этой главе описаны принцип работы программного продукта и его интерфейс.

3.1 Теоретическая часть программного продукта

Пиксель изображения в 24—х битном BMP формате занимает 3 Байта (24 бита) памяти (соответственно по 1 байту на каждый канал — Red, Green, Blue (RGB)).

Один символ текста занимает 1 Байт (8 бит) дискового пространства.

Если заменять байт изображения байтом текста, то мы получим абсолютно другой цвет пикселя, и как следствие, сильное искажение изображения.

Поэтому наиболее удобно заменять только 1 бит одного из каналов пикселя на 1 бит текста. Такая подмена будет незаметна для человеческого глаза. (При желании можно заменять последний бит каждого из каналов, алгоритм изменится незначительно).

Таким образом, один символ текста будет кодироваться в 8 пикселя изображения.

Возьмем пиксель со значениями кодируемого канала: Red = 255; BYTE R = 255; // 11111111. Нужно заменить один бит этого канала на бит символа, равный '0'. Если мы заменим первый бит Байта R, то получим: BYTE R = 11111110; // теперь R = 127 (отсчет идет справа налево, а не слева направо).

Теперь попробуем заменить последний бит пикселя: BYTE R = 01111111. Теперь R = 254, оттенок цвета изменится незначительно.

3.2 Принцип работы и интерфейс программного продукта в режиме шифрования

Основной режим работы программы предназначен для встраивания текстовой информации в изображение. В данном режиме пользователь имеет возможность устанавливать желаемые параметры встраивания и визуально оценивать качество полученного изображения.

1) Запуск программы.

При запуске, открывается основная форма программы (Рисунок 3.1).

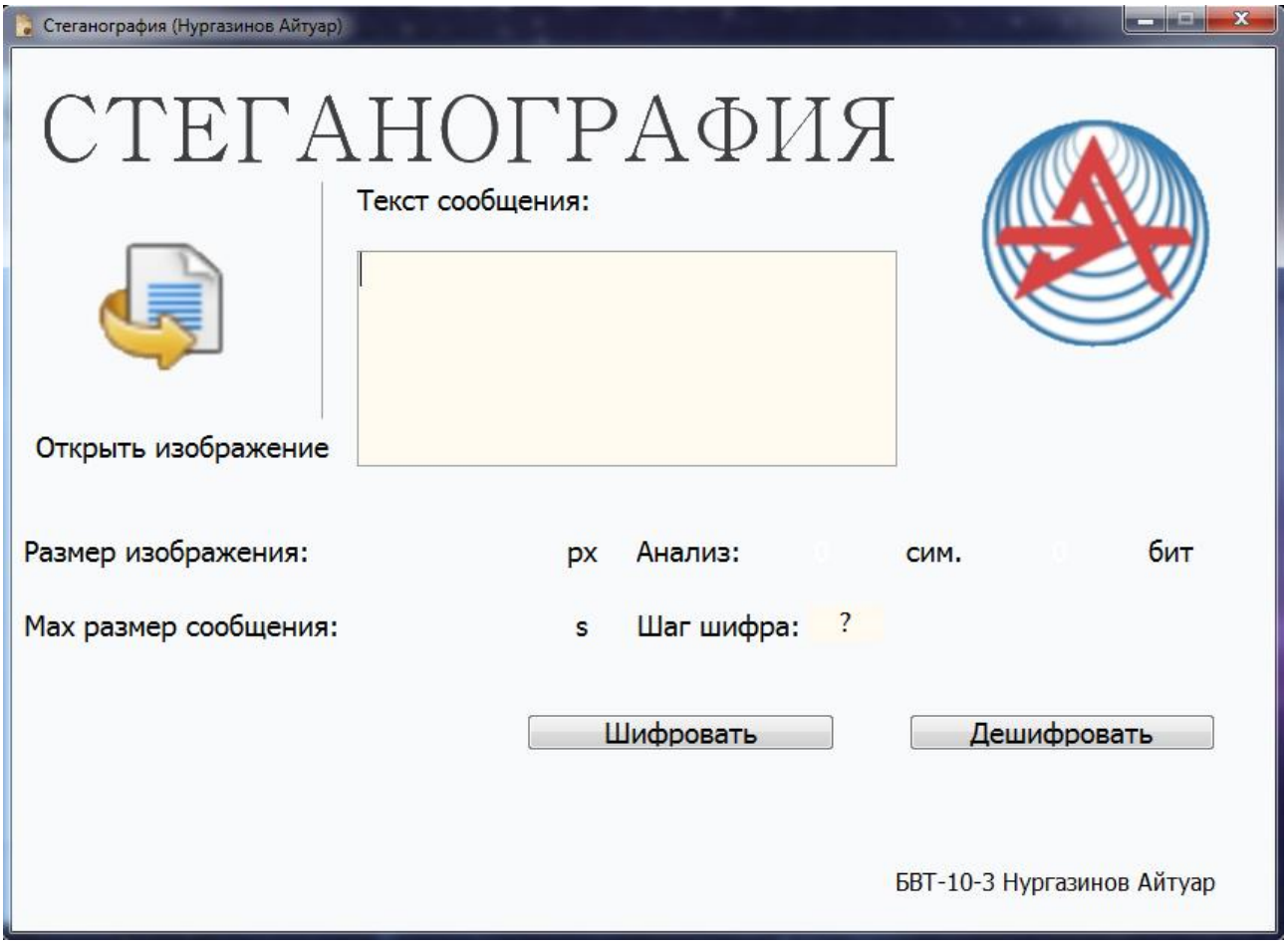

Рисунок 0.1 — Основная форма программы при начале работы.

2) Настройка параметров встраивания информации

Пользователю необходимо нажать на кликабельную картинку в левом верхнем углу программы и выбрать необходимое изображение на локальном диске которое он хочет использовать для шифрования (Рисунок 3.2).

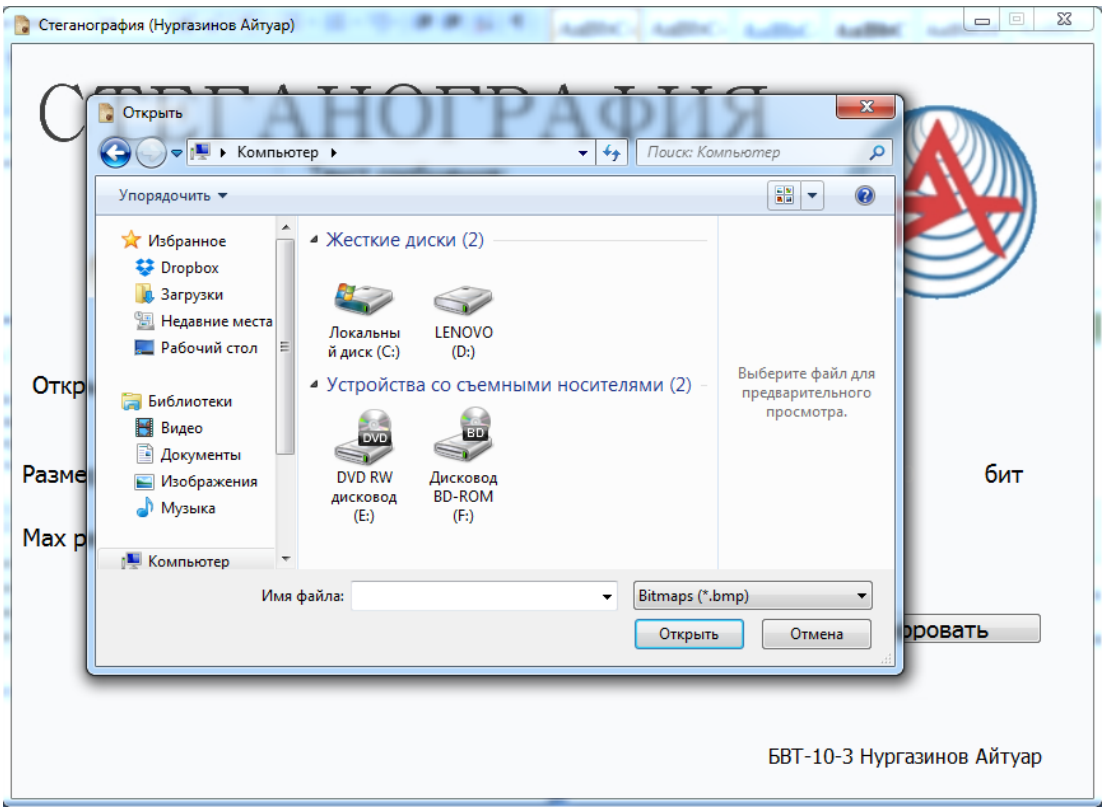

Рисунок 0.2 — Параметры встраивания изображения.

3) Определяется количество пикселей из которого состоит изображение (Рисунок 3.3).

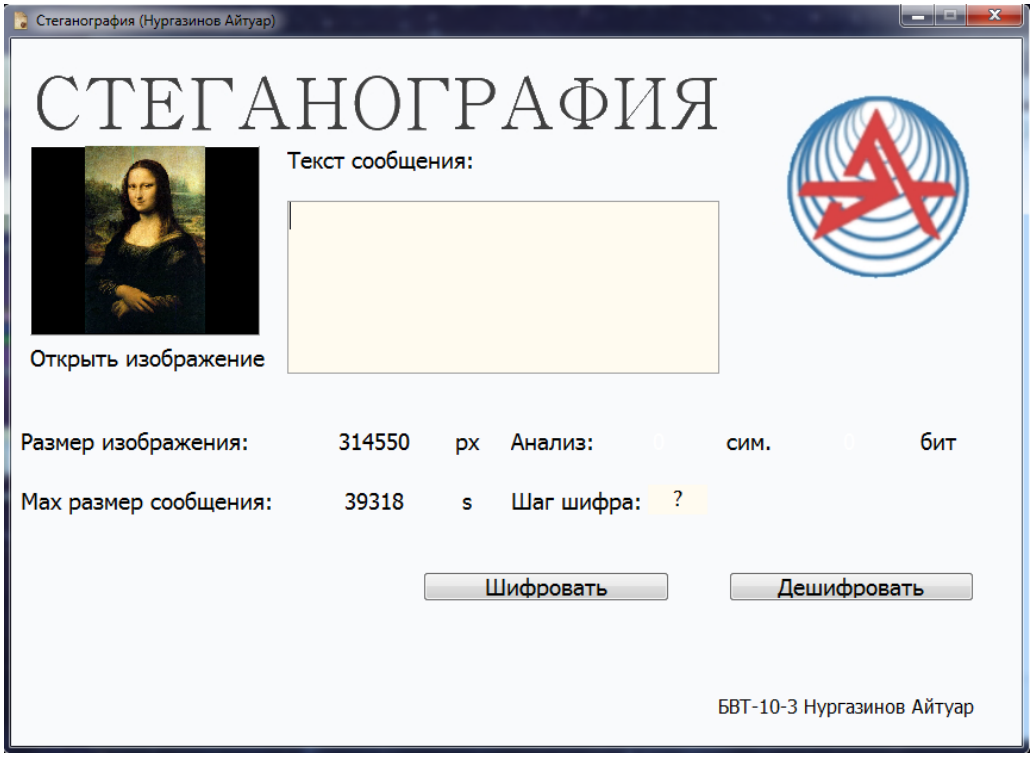

Рисунок 3.3 — Встраиваемое изображение

4) Вводим шифруемый текст (Рисунок 3.4).

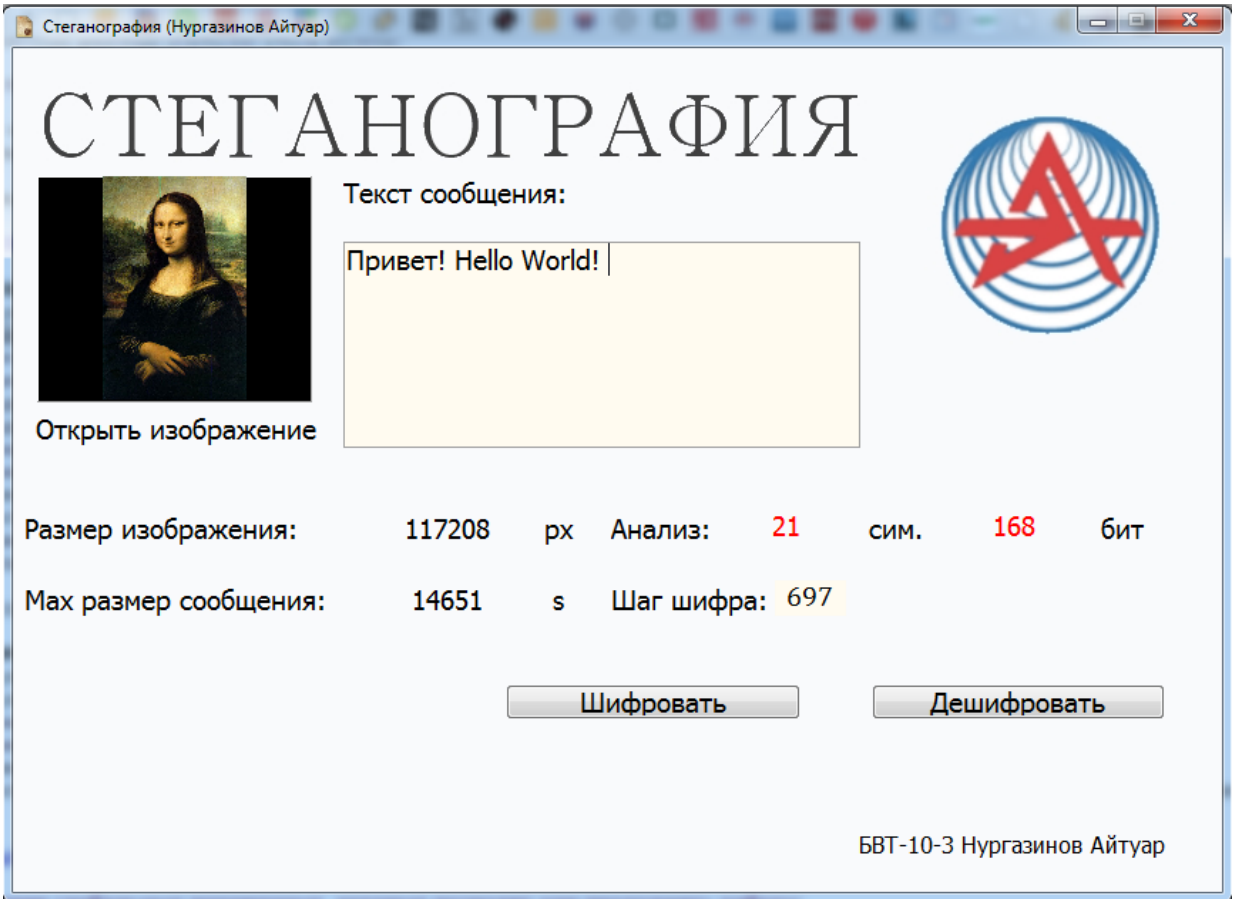

Рисунок 3.4 — Введенный текст

5) Значение делится на 8 и определяется максимальный размер текстового сообщения. Проверяем с максимально допустимым значением. По Битмапу отсчитывается шаг шифра, организуются два цикла:

А) Внешний: обход по символам;

Б) Внутренний: обход по битам символа.

Нажимаем кнопку «шифровать» (Рисунок 3.5).

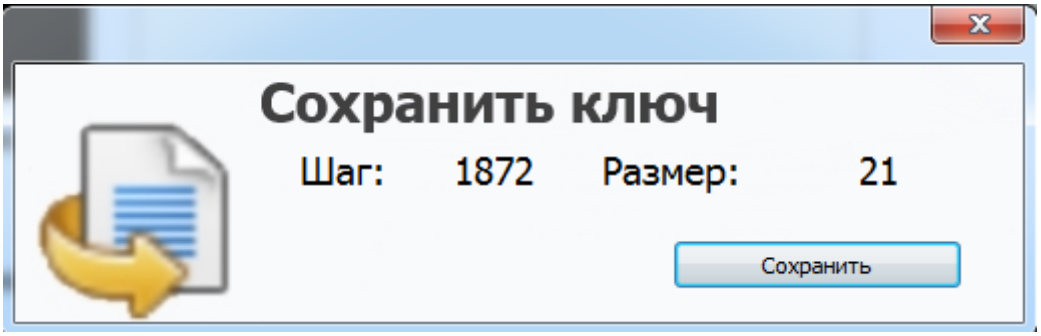

Рисунок 3.5 — Нажатие кнопки «шифровать»

6) Нажимаем кнопку «сохранить» и выбираем место в локальном диске (Рисунок 3.6).

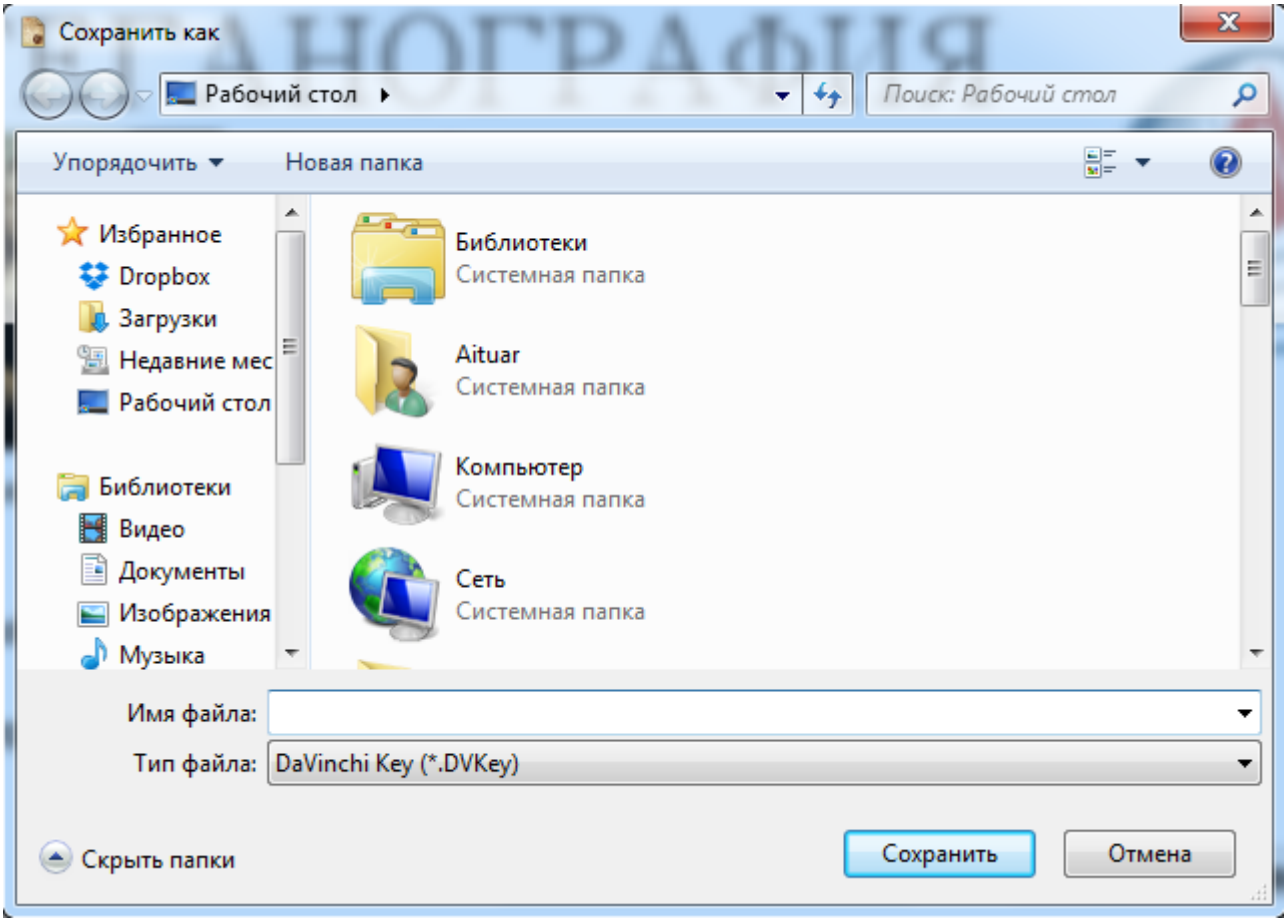

Рисунок 3.6 — Сохранение ключа шифрования

Ключ шифрования содержит две переменные:
А) шаг текста;
Б) размер текста.

3.3 Принцип работы и интерфейс программного продукта в режиме дешифрования

1) Запуск программы

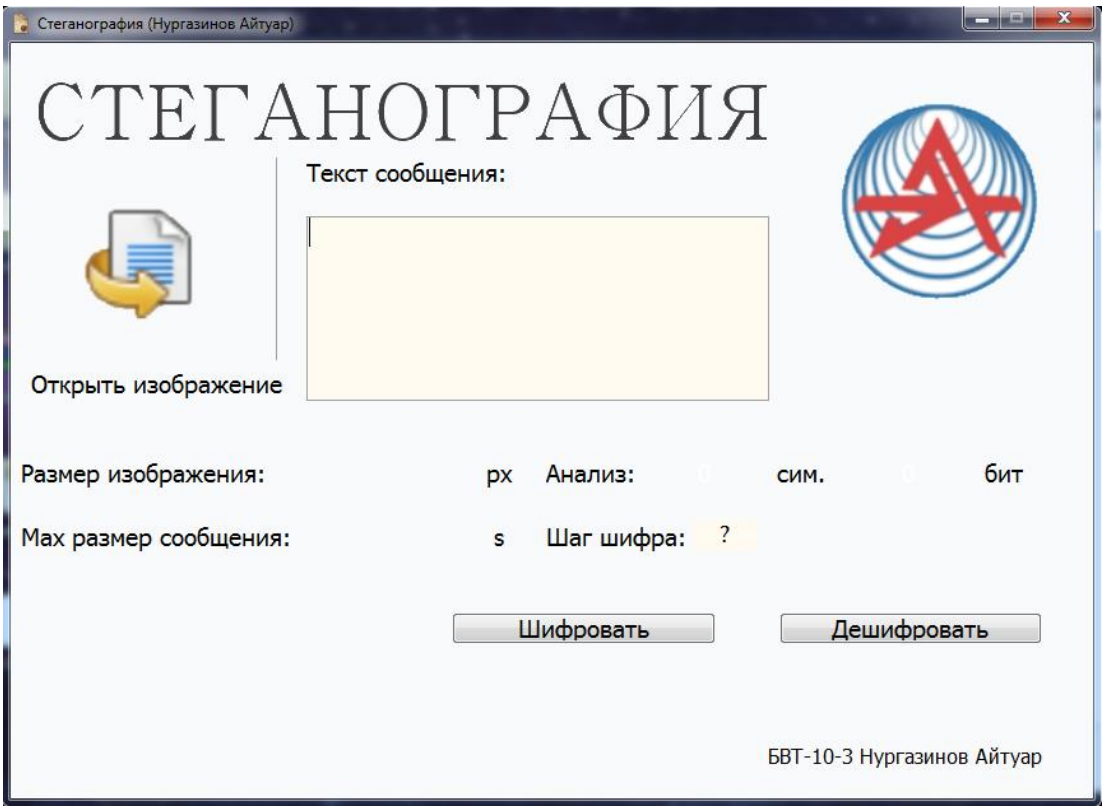

Рисунок 3.7 — Основная форма программы при начале работы.

2) Загружаем изображение

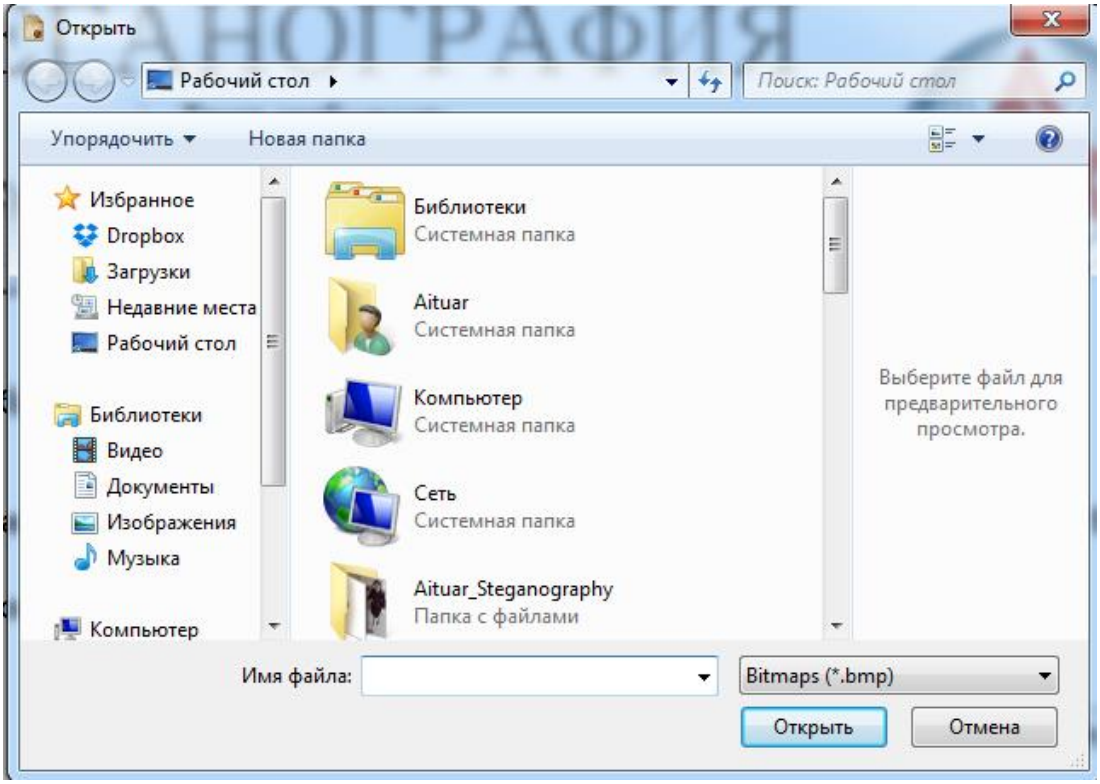

Рисунок 3.8 — Окно загрузки изображения

После того как выбранное изображение загружено, нажимаем кнопку дешифровать (Рисунок 3.9).

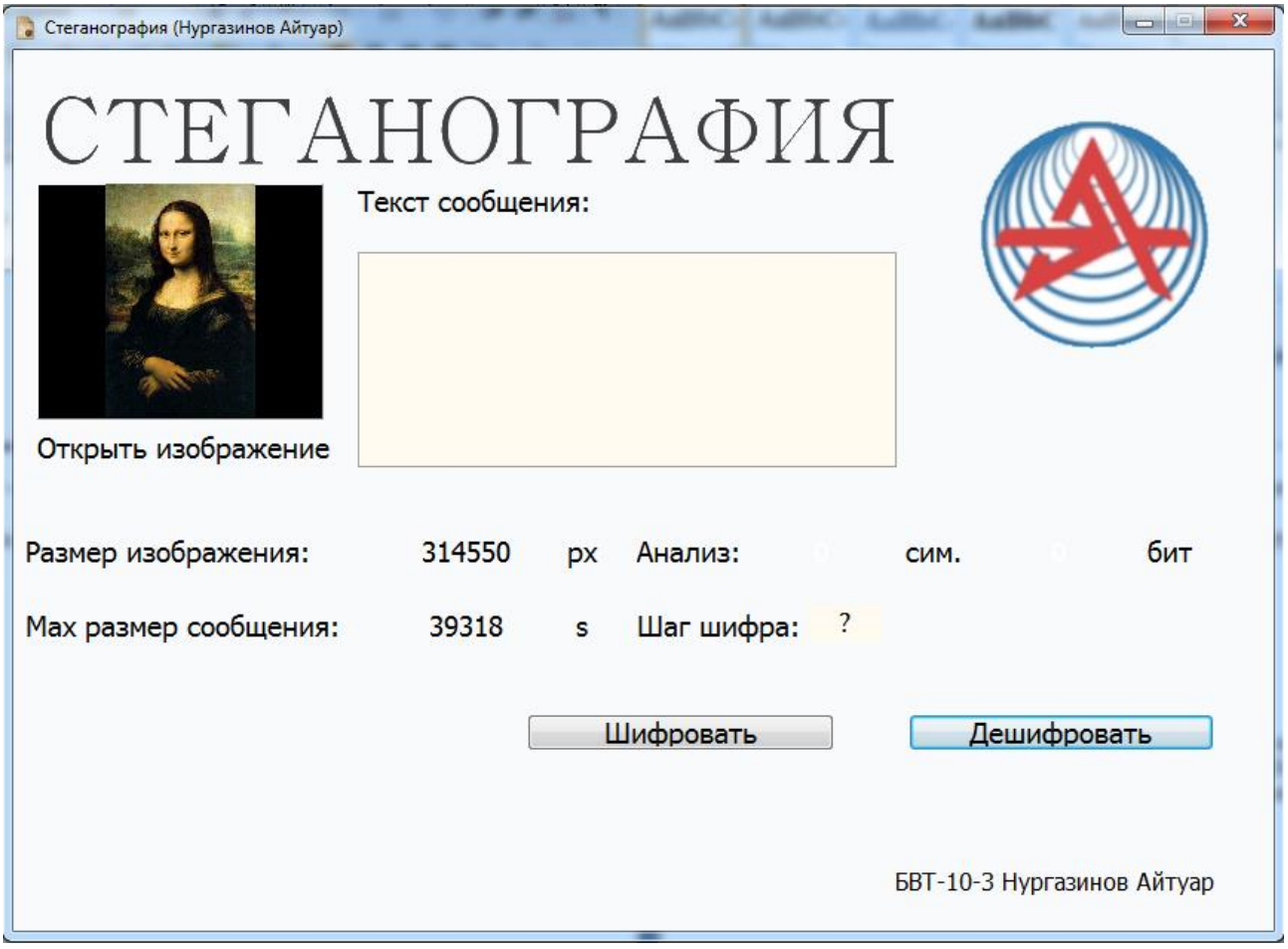

Рисунок 3.9 — Нажатие кнопки «дешифровать»

Далее выходит окно для поиска ключа шифрования сохраненного нами ранее (Рисунок 3.10).

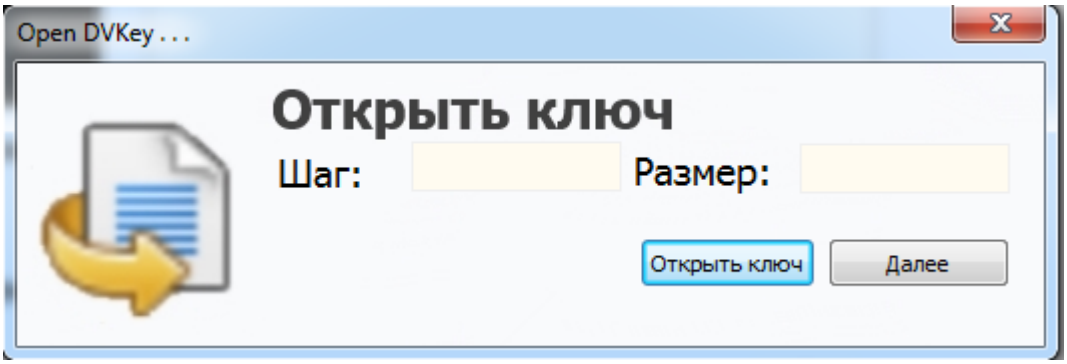

Рисунок 3.10 — Окно выбора ключа шифрования

Выбираем ключ шифрования и нажимаем «открыть» (Рисунок 3.11)

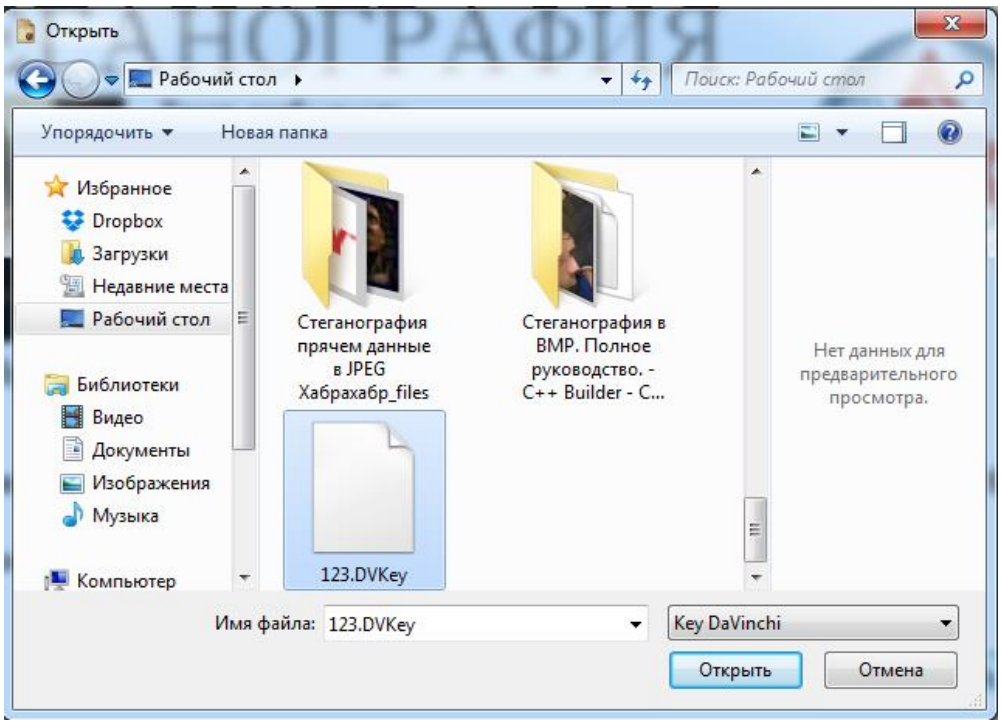

Рисунок 3.11 — Окно открытия ключа шифрования

Готово. Программа выходит на главный экран, и выводит сообщения зашифрованное ранее (Рисунок 3.12).

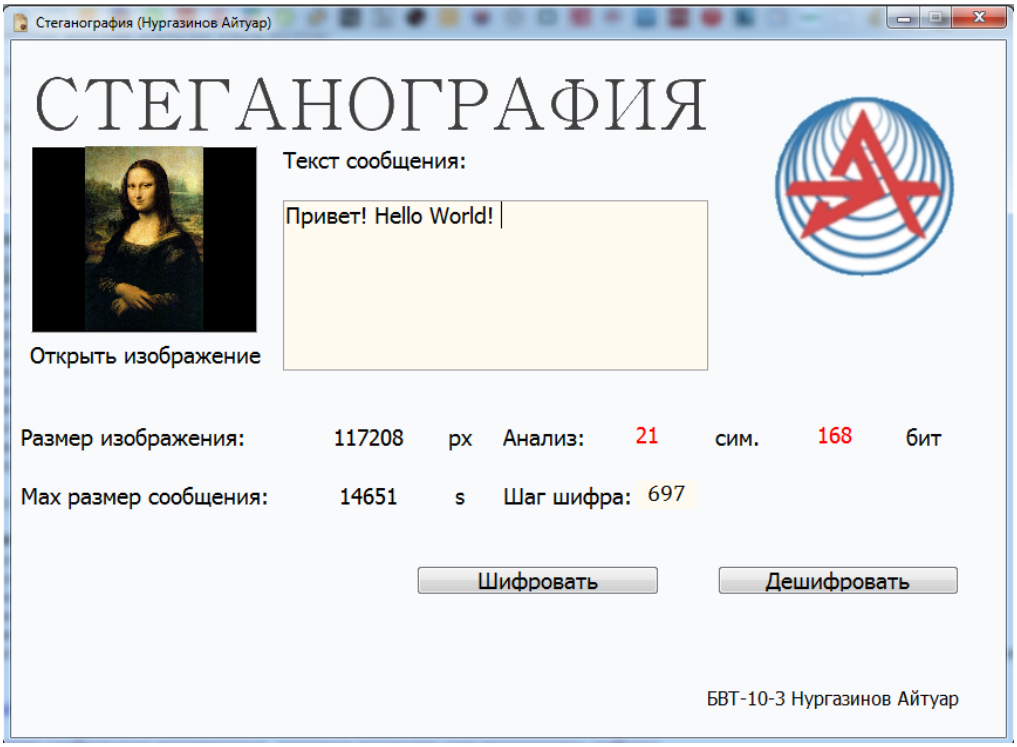

Рисунок 3.12 — Основная форма программы при полученном результате

4 Технико — экономическое обоснование

4.1. Описание работы и обоснование необходимости

Цель разработки данного ПП — является разработка программного продукта для стеганографического скрытия информации. Топологию следует разработать с учетом возможности принятия и передачи файлов по интернету, а также по локальной сети, при этом она должна обеспечивать должный уровень надежности и безопасности.

В данном разделе проводится рассмотрение экономической составляющей реализации данного проекта, отражающей временные, трудовые, и финансовые затраты на проект.

4.2 Расчет затрат на разработку информационных технологий

Под информационными технологиями понимаются экономические информационные системы (ЭИС), программные продукты (ПП), информационные базы данных и т.д.

Расчет полных затрат на разработку проектного решения в виде информационных технологий ($С_{пі}$) осуществляется по формуле

$$С_{пі} = З_{фот} + З_{сзі} + М_і + А + Р_{мі} + П_{зі} + Р_{ні} \quad (4.1)$$

где $З_{фот}$ — общий фонд оплаты труда разработчиков, тенге;

$З_{сзі}$ — отчисления по социальному налогу, тенге;

$М_і$ — затраты на материалы, тенге;

А — амортизация;

$Р_{мі}$ — затраты, связанные с эксплуатацией техники, тенге;

$П_{зі}$ — прочие затраты, тенге;

$Р_{ні}$ — накладные расходы, тенге.

Размер фонда оплаты труда разработчиков ($З_{ФОТ}$) рассчитывается по формуле

$$З_{ФОТ} = З_{оі} + З_{ді} \quad (4.2)$$

где $З_{оі}$ — основная заработная плата, тенге;

$З_{ді.}$ — дополнительная заработная плата, тенге.

Затраты на оплату труда зависят от объема и трудоемкости разработки программного обеспечения.

Общий объем (V_0) программного продукта определяется исходя из количества и объема функции, реализуемых программой

$$V_0 = \sum_{j=1}^{n} V_i \quad (4.3)$$

где V_i — объем отдельной функции ПО;
n — общее число функций.

Расчет уточненного объема ПО представлен в таблице 4.1.

Т а б л и ц а 4.1 — Перечень и объем функций программного модуля

№ функции	Наименование (содержание)	Объем функции (LOC)	
		по каталогу V_i	уточненный V_{yi}
201	Разработка схемы ПП	4300	1000
203	Создание ПП	2180	800
204	Заполнение ПП	2670	565
205	Создание процедур, триггеров, раздача прав	1260	622
206	Создание интерфейса программного продукта	6950	571
207	Разработка модуля ПП	9550	3890
208	Тестирование ПП	5480	1260
707	Отладка ПП	480	1292
	Итого:	32870	10000

Общая трудоемкость небольших проектов рассчитывается по формуле

$$Т_0 = Т_н * К_с * К_м * К_н \quad (4.4)$$

где $Т_н$ — нормативная трудоемкость;
$К_с$ — коэффициент, учитывающий сложность ПО;
$К_т$ — поправочный коэффициент, учитывающий степень использования при разработке стандартных модулей;
$К_н$ — коэффициент, учитывающий степень новизны ПО.

Посредством коэффициента сложности учитывается затраты труда, связанные со сложностью ПП (Таблица 4.2).

В разрабатываемом проекте K_i, за счет наличия у программного модуля одновременно трех характеристик:

1) режим работы в реальном времени;
2) управление удаленными объектами;
3) существенное распараллеливание вычислений.

Т а б л и ц а 4.2 — Дополнительные коэффициенты сложности ПО

Характеристика ПО	Значения $К_с$

1. Функционирование ПО в расширенной операционной среде (связь с другими ПО)	0,08
2. Интерактивный доступ	0,06
3. Обеспечение хранения, ведения и поиска данных в сложных структурах	0,07
4. Наличие у ПО одновременно нескольких характеристик по табл.Г4.1, приложение Г 4.1 2 характеристики 4.2 3 характеристики 4.3 Свыше 3—х характеристик	0,12 0,18 0,26

Коэффициент, учитывающий степень использования при разработке ПО стандартных модулей ($К_m$). Степень использования в разрабатываемом ПО стандартных модулей определяется их удельным весом в общем объеме проектируемого продукта. В данном проекте степень охвата реализуемых функций разрабатываемого ПО стандартными модулями, типовыми программами и ПО до 20%, следовательно $К_m$=0,9.

Поправочный коэффициент, учитывающий новизну разрабатываемого ПО ($К_н$) определяется на основе данных представленных в таблице 4.3 и составляет 1,0.

Т а б л и ц а 4.3 — Поправочные коэффициенты, учитывающие новизну ПО($К_н$).

Категория новизны	Степень новизны	Использование		Значение $К_н$
		На основе нового типа ПК	В среде новой ОС	
А	Принципиально новые ПО, не имеющие доступных аналогов	+ — + —	+ + — —	1,75 1,6 1,2 1,0

Нормативная трудоемкость ПО ($Т_н$) определяется на основе принятого в расчет V_y и категории сложности, которая уточняется с учетом сложности и новизны проекта и степени использования стандартных модулей при разработке.

В соответствии с этим, согласно укрупненным нормам времени на разработку ПО ($T_н$) в зависимости от уточненного объема ПО (V_0) и группы сложности (Приложение В) : объем ПО(строки исходного кода, LOC) 11000, категория сложности ПО 2—я — $T_н$ =291, категория сложности ПО 40.

Следовательно $Т_о$ будет равно

$$T_0 = 291 \cdot 0{,}12 \cdot 0{,}7 \cdot 1 = 24{,}44 \text{ (чел./дн.)}$$

Численность исполнителей проекта ($Ч_р$) рассчитывается по формуле

$$Ч_р = \frac{Т_о}{Т_р * Ф_{эф}} \quad (4.5)$$

где $Ф_{эф}$ − эффективный фонд времени работы одного работника в течение года (дн.);

$Т_о$ − общая трудоемкость разработки проекта (чел./дн.);

$Т_р$ − срок разработки проекта (лет).

Срок разработки проекта ($Т_р$) определяется по формуле

$$Т_р = \frac{Т_о}{Ч_р * Ф_{эф}} \quad (4.6)$$

где $Ч_р$ — плановое число разработчиков.

Эффективный фонд времени работы одного работника ($Ф_{эф}$) рассчитывается по формуле

$$Ф_{эф} = Д_г — Д_п — Д_в — Д_о \quad (4.7)$$

где $Д_г$ − количество дней в году;

$Д_п$ − количество праздничных дней в году;

$Д_в$ − количество выходных дней в году;

$Д_о$ − количество дней отпуска.

В соответствии производственным календарем на 2014 год.: $Д_г$ − 365; $Д_п$ = 14 ; $Д_в$ — 103; $Д_о$ — 10, то по формуле (6.7) получим

$Ф_{эф}$ = 365 — 14 — 103 — 10 = 228 дней

Плановое число разработчиков $Ч_р$ = 1, следовательно по формуле (4.6)

$Т_р$ = 24,44 / (1 * 228) = 0,11 года = 40 дней

Таким образом, согласно произведенным расчетам и в соответствие с формулой (4.5)

Ч = 24,44 / (0,11 * 228) = 1 чел.

Основная заработная плата исполнителей на конкретное ПО рассчитывается по формуле

$$З_{оі} = \sum_{i=1}^{n} Т_{чі} \times Т_ч \times Ф_п \times К \quad (4.8)$$

где n − количество исполнителей, занятых разработкой конкретного ПО;

$Т_{чі}$ − часовая тарифная ставка і—го исполнителя (тыс.тенге);

$Ф_п$ − плановый фонд рабочего времени і—го исполнителя (дней);

$Т_ч$ − количество часов работы в день (час), 8 часов;

К − коэффициент премирования, составляет 1,2.

По данным о специфике и сложности выполняемых функций составляется штатное расписание группы специалистов—исполнителей, участвующих в разработке ПО, с определением образования, специальности, квалификации и должности (Таблица 4.4).

Т а б л и ц а 4.4 — Сведения по работникам, задействованным в проекте

Специалист —исполнитель	Количество, человек	Заработная плата в месяц, тенге
Инженер—программист	1	170 000
Итого		170 000

Часовая тарифная ставка рассчитывается путем деления месячной тарифной ставки на установленную при 40—часовой недельной норме рабочего времени расчетную среднемесячную норму рабочего времени в часах ($Ф_р$)

$$Т_ч = \frac{Т_н}{Ф_р} \quad (4.9)$$

где $Т_ч$ − часовая тарифная ставка (тыс.тенге);

$Т_м$ − месячная тарифная ставка (тыс.тенге).

Таким образом

$$Т_ч = \frac{170\,000}{168} = 1011{,}4 \text{ тенге/час}$$

По формуле (4.8) можно рассчитать основную заработную плата исполнителей

$З_{оі}$ = 1011,4 * 8 * 40 * 1,38 = 446 634 тенге

Дополнительная заработная плата составляет 10% от основной заработной платы и рассчитывается по формуле

$$З_{дi} = З_{оi} \times Н_д/100$$

(4.10)

где $Н_д$ — коэффициент дополнительной заработной платы разработчиков 22%.

$З_{дi}$ = 446 634 × 0,24 = 107 192 тенге

Социальный налог составляет 11% (ст. 358 п. 1 НК РК) от дохода работника, и рассчитывается по формуле

$$З_{сзi} = (ФОТ — ПО) \times 11\%$$

(4.11)

где ПО — пенсионные отчисления, которые составляют 10% от ФОТ и социальным налогом не облагаются

$$ПО = ФОТ \times 10\%$$

(4.12)

Таким образом

ПО= 446 634 * 0,1 = 44 663 тенге,

$З_{сзi}$ = (446 634 — 44 663) * 0,11 = 40 197 тенге

Затрат на материалы определяются по формуле

$$М_i = (З_{осн.} \times Н_{мз})/ 100\%$$

(4.13)

где $Н_{мз}$ − норма расхода материалов от основной заработной платы (3—5%).

$М_i$ = 446 634 * 0.042 = 18 758 тенге

Амортизационные отчисления производятся по установленным нормам амортизации, выражаются, в процентах к балансовой стоимости оборудования и рассчитываются по формуле

$$A = \frac{С_{обор} * Н_а * N}{365 * 100} \quad (4.14)$$

где $Н_а$ — норма амортизации (25 %);

$С_{обор.}$ — первоначальная стоимость оборудования;

N — фактический срок эксплуатации оборудования, 40 дней;

Данные по стоимости оборудования представлены в таблице 4.5.

Т а б л и ц а 4.5 —Оборудование необходимое для разработки ПП

Наименование	Цена с НДС, тенге	Цена без НДС, тенге
Монитор	32 000	28 160
Материнская плата	13 000	11 264
Процессор	30 000	26 400
Видеокарта	23 000	20 240
HDD	12 000	10 560
DVD—RW	6 500	5 720
CPU Cooler	5 500	4 840
Оперативная память	8 000	7 040
Клавиатура	1 000	880
Мышь	1 000	880
Принтер	8 000	7 040

$С_{обор.}$ = 140 000 тенге

Тогда, согласно формуле (4.14)

$$A = \frac{140\,000 \cdot 25 \cdot 60}{365 \cdot 100} = 5753{,}4$$

Расходы по статье «Машинное время» ($Р_{мi}$) включают оплату машинного времени, необходимого для разработки и отладки ПО, которое определяется по нормативам (в машино—часах) на 100 строк исходного кода ($Н_{мв}$) машинного времени в зависимости от характера решаемых задач и типа ПК

$$Р_{мi} = Ц_{мi} \times (V_{oi}/100) \times Н_{мв} \qquad (4.15)$$

где $Ц_{мi}$ − цена одного машино—часа (тыс.тенге);

V_{oi} − общий объем ПО (строк исходного кода);

$Н_{мв}$ − норматив расхода машинного времени на отладку 100 строк исходного кода (машино—часов).

Норматив расхода машинного времени на отладку 100 строк машинного кода определяется на основе данных представленных в приложении Д (Оценка значений среднего машинного времени отладку 100 строк исходного кода без применения ПО) и составляет 12.

Таким образом

$Р_{Мі}$ = 1 014,2 * (11 000 / 100) * 12 = 1 338 048 тенге

Расходы по статье «Прочие затраты» ($П_{зі}$) на конкретное ПО включают затраты на приобретение и подготовку специальной научно—технической информации и специальной литературы. Определяются по нормативу, разрабатываемому в целом по организации, в процентах к основной заработной плате

$$П_{зі} = З_{оі} \times Н_{пз}/100 \quad (4.16)$$

где $Н_{пз}$ − норматив прочих затрат в целом по организации в (%), в дипломной работе нужно брать 30% .

Подставляем все данные в формулу (4.20) получаем

$П_{зі}$ = 446 634 * 0,2 = 89 326 тенге

Затраты по статье «Накладные расходы» ($Р_{ні}$), связанные с необходимостью содержания аппарата управления, вспомогательных хозяйств и опытных(экспериментальных) производств, а также с расходами на общехозяйственные нужды ($Р_{ні}$), относятся на конкретное ПО по нормативу ($Н_{рн}$) в процентном отношении к основной заработной плате исполнителей. Норматив устанавливается в целом по организации

$$Р_{ні} = З_{оі} \times Н_{рн}/100\% \quad (4.17)$$

где $Р_{ні}$ — накладные расходы на конкретную ПО (тыс.тенге);

$Н_{рн}$ — норматив накладных расходов в целом по организации в (%), в дипломной работе нужно брать 70%.

Подставляем все данные в формулу (4.17) получаем

$Р_{ні}$ = 446 634 * 0,7 = 312 643 тенге,

$С_{пі}$ = 1 843 061,8 тенге

Результаты выполненных расчетов представлены представлены в таблице 4.6 и на рисунке 4.1

Т а б л и ц а 4.6 — Себестоимость разработки ПП для стеганографического скрытия информации

Затраты на разработку	Условное обозначение	Значение, тенге	В процентах от общей суммы
Фонд оплаты труда	$З_{ФОТ}$	446 634	12,7
Социальный налог	$З_{сзі}$	107 192	1,27
Материалы	$М_і$	18 758	0,53
Амортизация	А	5753,4	0,31
Машинное время	$Р_{мі}$	1 338 048	72,44
Прочие затраты	$П_{зі}$	89 326	3,82
Накладные расходы	$Р_{ні}$	312 643	8,91
Итого:		1 843 061,8	100,00

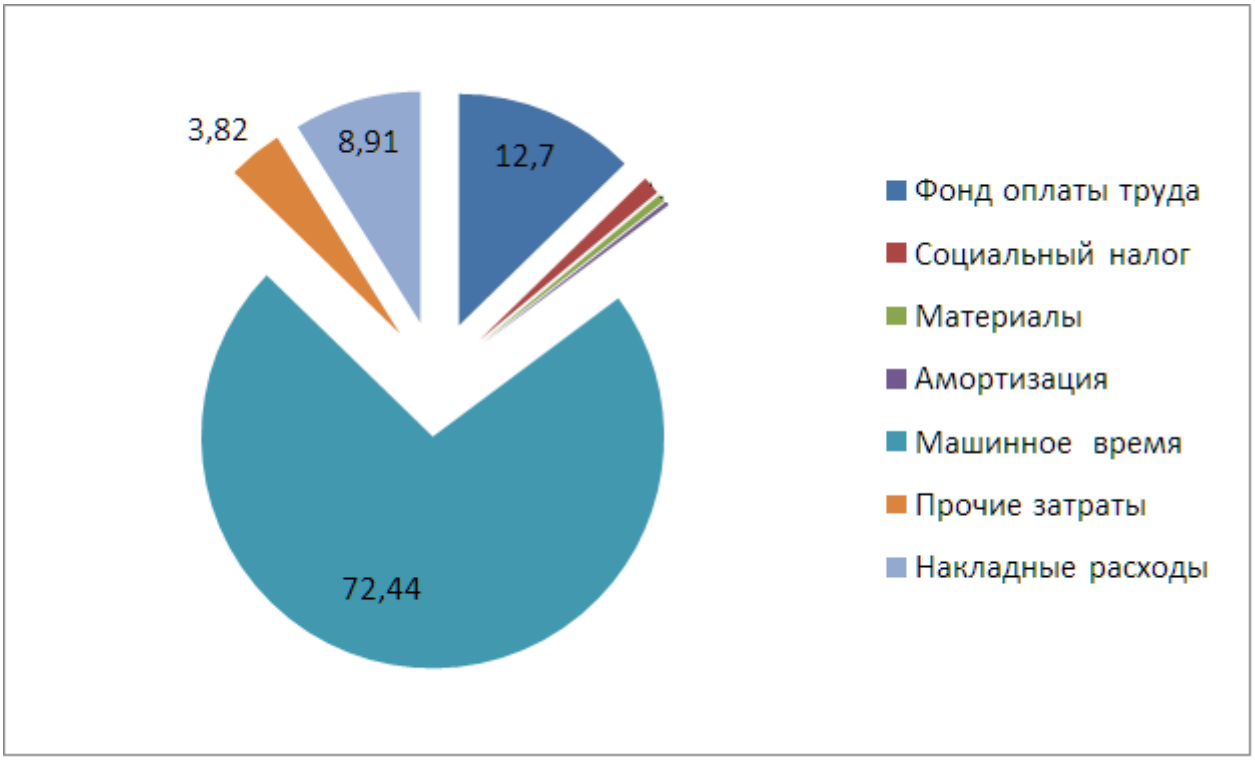

Рисунок 4.1 — Структура себестоимости разработки ПП для стеганографического скрытия информации

4.3 Расчет цены программного продукта

Расчет цены ПП в организационно—экономической части дипломной работы предлагается производить следующим образом:

1) если ПП разработан одной организаций по заказу другой и не предназначен для тиражирования, то затраты на разработку ПП считаются его себестоимостью, и при формировании цены применяется затратный метод;

2) если ПП предназначен для тиражирования, то конечная цена определяется путем экспертных оценок на основании ценностного подхода с учетом текущих цен конкурентов (если существуют аналогичные ПП).

Расчет цены ПП, который разработан одной организаций по заказу другой и не предназначен для тиражирования, осуществляется по формуле

$$Ц_{ПП} = З_{РПР} + П_{п} + НДС \quad (4.18)$$

где $Ц_{ПП}$ — цена программного продукта, тенге;

$З_{РПР}$ — затраты на разработку проектного решения, в данном случае программного продукта, тенге;

$П_{п}$ — планируемая прибыль, тенге;

$НДС$ — налог на добавленную стоимость, тенге.

Планируемая прибыль рассчитывается по формуле

$$П_{п} = З_{РПР} + R_{НПП} \quad (4.19)$$

где $R_{НПП}$ — нормативная рентабельность ПП, определяемая организацией.

НДС— начисленный на ПП, определяется следующим образом

$$НДС = (З_{РПР} + П_{п})* k_{НДС} \quad (4.20).$$

где $k_{НДС}$ — ставка налога на добавленную стоимость.

Подставляем все значения в формулы (4.19) — (4.20) и получаем по формуле (4.20) учитывая, что $З_{РПР} = С_{пi}$, $R_{НПП}$ — это процент рентабельности по отношению к себестоимости составляет 20%

Подставив данные в формуле (4.21) получаем

НДС = (1 843 061,8 + 2 211 674,16) * 0,12 = 486 568,32 тенге

Подставив данные в формуле (4.22) получаем

$Ц_{ПП}$ = 1 843 061,8 + 2 211 674,16 + 486 568,32 = 4 541 304,28 тенге

Заключение

В настоящее время развиваются методы компьютерной стеганографии — самостоятельного научного направления информационной безопасности, изучающей проблемы создания компонентов скрываемой информации в открытой информационной среде, которая может быть сформирована вычислительными системами и сетями. Особенностью стеганографического подхода является то, что он не предусматривает прямого оглашения факта существования защищаемой информации. Это обстоятельство позволяет в рамках традиционно существующих информационных потоков или информационной среды решать некоторые важные задачи защиты информации ряда прикладных областей.

В результате выполнения дипломного проектирования были решены следующие задачи:

Описаны наиболее популярные методы встраивания информации в пространственную область изображения. Приведен их сравнительный анализ и выбран наиболее стойкий к пассивным атакам метод, который позволяет передавать большое количество информации.

Разработан программный продукт на C++ Builder 2009, для скрытой передачи информации в пространственной области изображения.

Список используемой литературы

1. Артёхин Б.В. Стеганография // Журнал "Защита информации. Конфидент", 1996. — №4. — С.47—50.
2. Швидченко И.В. Анализ криптостеганографических алгоритмов // Проблемы управления и информатики, 2007. — № 4. — С. 149—155.
3. Барсуков В.С., Романцов А.П. Компьютерная стеганография: вчера, сегодня, завтра. — М.: Солон—Пресс, 2005. — 150 с.
4. Генне О.В. Основные положения стеганографии // Журнал "Защита информации. Конфидент". — 2000. — №3.
5. Грибунин В.Г., Оков И.Н., Туринцев И.В. Цифровая стеганография. — М.: Солон—Пресс, 2002. — 272 с.
6. Конахович Г.Ф., Пузыренко А.Ю. Компьютерная стеганография Теория и практика. МК—Пресс, 2006. — 288 с.
7. Грибунин В.Г. Критерии оценки надёжности паролей. — М.: РУСКАРД, 2003.
8. Игнатов В.А. Теория информации и передачи сигналов. — М.: Радио и связь, 1991. — 280с.
9. Кустов В.Н., Федчук А.А. Методы встраивания скрытых сообщений // Журнал "Защита информации. Конфидент". 2000. — №3. — С.34.
10. Аграновский А.В., Девянин П.Н., Хади Р.А., Черемушкин А.В. Основы компьютерной стеганографии. — М.: Радио и связь, 2003. — 152 с.
11. Барсуков В.С. Стеганографические технологии защиты документов, авторских прав и информации // Обзор специальной техники. — 2000. — №2. — С.31 — 40.
12. Васильев А.В. Технико—экономическое обоснование дипломных проектов (работ). — СПб: ГЭТУ, 2002.
13. Долин П.А. Справочник по технике безопасности. — М.: Энергоиздат, 1982.— 342 с.
14. Каменев П.Н. Отопление и вентиляция. Часть II Вентиляция. — М.: Издательство литературы по строительству, 1966.
15. ГОСТ 12.1.005—88. Санитарно—гигиенические требования к воздуху рабочей зоны. – М.: Издательство стандартов, 1988.

Приложение А

```
#include <vcl.h>
#include <math.h>
#include <string.h>
#pragma hdrstop

#include "Steganografia.h"
#include "Save.h"
#include "OpenKey.h"
//---------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TStego *Stego;
//---------------------------------------------------
fastcall TStego::TStego(TComponent* Owner)
     : TForm(Owner)
{
}
//---------------------------------------------------


//+++ОПИСАНИЕ ГЛОБАЛЬНЫХ ПЕРЕМЕННЫХ +++//

     Graphics::TBitmap  *Bmp;  //  переменная  для  хранения  BMP
изображения

     TPicture *Warning;

     int  ImageMaxHeight;  //  максимальные  размеры  отображения
изображения
     int ImageMaxWidth;
     int WIDTH;
     int HEIGHT;
     float koefX;      // коеффициенты сжатия изображения
     float koefY;
     bool  ERR_OPEN_BMP  =  true;   //  переменные  для  отслеживания
ошибок
     bool ERR_OPEN_KEY = true;
     bool ERR_NO_TEXT = true;
     bool ERR_TEXT_LENGTH = false;

     unsigned int IMAGE_SIZE = 0;
     unsigned int TEXT_SIZE = 0;
     unsigned int MAX_TEXT_SIZE = 0;

     //++++++++++++++++++++++++++++++++++//


     // #### ФУНКЦИИ ### //

     TPoint GetPosition(TPoint Point, int Shag, int Width)
```

Продолжение приложения А

```
     /* Функция оперделяет следующую координату пиксела, который
будет кодироваться
        TPoint Point — предыдущая координата
        Shag — шаг кодирования
        Width — ширина Битмапа
     */
     {  TPoint REZ;
        REZ.y += Point.y;

     int BUF = Width — Point.x — Shag;

     if (BUF <= 0) {
                          REZ.y++;  BUF = abs(BUF);
                          while (BUF >= Width)
                          {BUF —= Width; REZ.y++;}
                       }
                       else BUF = REZ.x + Width — BUF;
     REZ.x = BUF;

     return REZ;
     }

     int *ByteToBinary (BYTE val)
     /*  Разбивает исходный Байт на биты. Возвращает массив из
восьми элементов
        Каждый Nй элемент соответствует Nму биту.
        Принимает значения 1 или 0
     */
     {           int *mass = new int[8];
                    int t, i;
                    for (t = 128, i = 0; t > 0; t /= 2, i++)
                    {
                         if ((val & t) != 0) mass[i] = 1;
                         else if ((val & t) == 0) mass[i] = 0;
                    }
       return mass;
     }

     BYTE BinaryToByte (int *mass)
     /* Собирает из массива битов один целый байт и возвращает
его.
        Входной   параметр:   массив   из   восьми   элементов,
соответсвтующих
        битам Байта
     */
     {
                    BYTE Mask   = 00000001;
                    BYTE Result = 00000000;
                    BYTE Mask2;
```

Продолжение приложения А

```
                    int j = 0;
                    for (int i = 7; i > -1; i--, j++)
                    {
                      if (mass[i] == 1)  {Mask2 = (Mask << (j));
                                          Result          =
Result|Mask2;
                                          }
                    }
                    return Result;
     }

     int BinaryToDec (BYTE val)
     /* Переводит Байт из двоичной системы в десятичную
     */
     {   int Result = 0;

          for (int t = 1, i = 0; t < 129; t *= 2, i++)
                    {
                         if ((val & t) != 0) Result += pow(2,i);
                    }
      return Result;
     }

     int GetBitValue(BYTE B, int N)
     /* Получает значение N-ого бита в байте B
        Возвращает 1 или 0 (в зависимости от значения Бита)
     */
     { int k = 256;

            for (int i = 0; i < N; i++) k /= 2;

            if ((B & k) != 0) return 1;
            else return 0;
     }

     BYTE ReadBitToByte (int Bit, BYTE B)
     /* Записывает в байт B на последнюю позицию бит Bit
     */
     {
           BYTE A = 00000001;
           BYTE Result = B;

           if (Bit == GetBitValue(B,8)) return B;
               else if (Bit == 1) return Result = Result|A;
                     else if (Bit == 0) return B - A;

                     return NULL;
     }

     // ########################################### //
```

Продолжение приложения А

```
     void __fastcall TStego::FormCreate(TObject *Sender)
```

```
{
Warning = Image1->Picture;
ImageMaxHeight = PanelImage->Height-2;
ImageMaxWidth = PanelImage->Width-2;
Label3->Caption = "";
Label4->Caption = "";
Label9->Caption = "";
Label12->Caption = "";

RichEdit1->Clear();
}
//---------------------------------------------------------------------------

void __fastcall TStego::Image1DblClick(TObject *Sender)
{
// ------- Открываем изображение ------- //
OpenPictureDialog1->Execute();
if (OpenPictureDialog1->FileName == "")
            {
                    Label3->Caption = "";
                    Label4->Caption = "";
                    ERR_OPEN_BMP = true;
                    return;
            }

Image1->HelpKeyword = OpenPictureDialog1->FileName;

// ------- отображаем на экран ------- //
Image1->Picture->LoadFromFile(OpenPictureDialog1->FileName);

WIDTH  = Image1->Picture->Width;
HEIGHT = Image1->Picture->Height;

koefY = (float) WIDTH/ImageMaxWidth;
koefX = (float) HEIGHT/ImageMaxHeight;

if (koefY > koefX)
      {
                Image1->Height = HEIGHT / koefY;
                Image1->Width = ImageMaxWidth;
                Image1->Left = 0;
                Image1->Top   =   (PanelImage->Height-Image1-
>Height)/2;
      }
else
      {
                Image1->Width = WIDTH / koefX;
                Image1->Height = ImageMaxHeight;
                Image1->Top = 0;
```

Продолжение приложения А

```
                Image1->Left   =   (PanelImage->Width-Image1-
>Width)/2;
```

```
          }

     if  (!Image1->Stretch)  Image1->Stretch  =  true;          //
подгоняем по размеру

                Bmp = new Graphics::TBitmap;
                Bmp->LoadFromFile(OpenPictureDialog1->FileName);
// копируем изображение в Bmp
                Bmp->PixelFormat  =  pf24bit;  //  устанавливаем
глубину цвета в 24 bit (стандарт без Альфа канала)  RGB888

     IMAGE_SIZE = WIDTH*HEIGHT;
     MAX_TEXT_SIZE = IMAGE_SIZE/8;

     Label3->Caption =  IntToStr((int)IMAGE_SIZE);
     Label4->Caption =  IntToStr((int)MAX_TEXT_SIZE);

     ERR_OPEN_BMP = false;
     RichEdit1->OnChange(this);
     }
     //---------------------------------------------------------------

     void __fastcall TStego::RichEdit1Change(TObject *Sender)
     {
     TEXT_SIZE = RichEdit1->Text.Length();

     Label9->Caption = IntToStr((int) TEXT_SIZE);
     Label12->Caption = IntToStr((int)(TEXT_SIZE * 8));

     if (TEXT_SIZE != 0) ERR_NO_TEXT = false;
       else ERR_NO_TEXT = true;

     if ((TEXT_SIZE != 0)&&(ERR_OPEN_BMP == false))
          {
             Edit1->Text                                          =
IntToStr((int)(MAX_TEXT_SIZE/TEXT_SIZE));
          }
          else
             {
                Edit1->Text = "?";
             }
     if (TEXT_SIZE > MAX_TEXT_SIZE)
          {
             ERR_TEXT_LENGTH = true;
             Label9->Font->Color = clRed;
             Label12->Font->Color = clRed;
          }
          else
```

Продолжение приложения А

```
                ERR_TEXT_LENGTH = false;
                Label9->Font->Color = clWhite;
                Label12->Font->Color = clWhite;
```

```
            }
        }
        //---------------------------------------------------------------------
        void __fastcall TStego::Button1Click(TObject *Sender)
        {
        if       ((ERR_OPEN_BMP    ==    true)||(ERR_NO_TEXT       ==
true)||(ERR_TEXT_LENGTH == true))
              {
                  char *str = new char[512];
                  strset(str, '\0');
                  if (ERR_OPEN_BMP == true) strcat(str,"Ошибка 001:
изображение не найдено.\n");
                  if (ERR_NO_TEXT  == true) strcat(str,"Ошибка 002:
не введен текст.\n");
                  if  (ERR_TEXT_LENGTH  ==  true)  strcat(str,"Ошибка
003: текст слишком длинный.\n");
                  MessageBox(NULL,    str,    "Ошибка!",    MB_OK    |
MB_ICONERROR);
                  return;
              }

        int Shag = StrToInt(Edit1->Text);
        char   *String   =   new   char[RichEdit1->Text.Length()+1];//
создаем переменную для хранения кодируемого текста
        strcpy(String,RichEdit1->Text.c_str());  //  копируем  в  нее
кодируемый текст

         TPoint Next;     // позиция следующего кодируемого пиксела
         TPoint First;    // позиция кодированного пиксела
         BYTE TextByte;   // переменная для хранения символа

        //------------------- ШИФРУЕМ СООБЩЕНИЕ ---------------------//

         for (int i = 0; i < TEXT_SIZE; i++)  // перебор символов
             {
                   TextByte   =   String[i];   //   загоняем   очередной
символ

                   for  (int  j  =  0;  j  <  8;  j++)  //  перебор  битов
символа
                       {
                        Next = GetPosition(First, Shag, WIDTH);

                        TColor       COLORR       =       Bmp->Canvas-
>Pixels[First.x][First.y];

                        BYTE R=GetRValue(COLORR);     //
```

Продолжение приложения А

```
                        BYTE  G=GetGValue(COLORR);        //  получили
каналы
                        BYTE B=GetBValue(COLORR);     //
```

```
                    int bit = GetBitValue(TextByte, j+1); //
получили записываемый бит
                    R = ReadBitToByte (bit, R);             //
записали jй бит в канал

                    Bmp->Canvas->Pixels[First.x][First.y]      =
RGB(R,G,B);  // переопределили цвет

                    First = Next;
                   }
          }
     //--------------------- ЗАШИФРОВАЛИ --------------------------//

     Form1->Label9->Caption = Shag;
     Form1->Label12->Caption = TEXT_SIZE;

     SavePictureDialog1->Execute();
     if     (SavePictureDialog1->FileName     !=     "")     Bmp-
>SaveToFile(SavePictureDialog1->FileName);
      else
        {
          MessageBox(NULL,    "Изображение    не    сохранено...",
"Ошибка!", MB_OK | MB_ICONERROR);
           return;
        }
     Form1->Left = Stego->Left + Stego->Width/2 - Form1->Width/2;
     Form1->Top   =    Stego->Top   +   Stego->Height/2   -   Form1-
>Height/2;
     Form1->ShowModal();

     }
     //---------------------------------------------------------------

     void __fastcall TStego::Button2Click(TObject *Sender)
     {
     if  (ERR_OPEN_BMP == true)
           {
               MessageBox(NULL,   "Ошибка   001:   изображение   не
найдено.\n", "Ошибка!", MB_OK | MB_ICONERROR);
               return;
           }

     Open->Left = Stego->Left + Stego->Width/2 - Open->Width/2;
     Open->Top =  Stego->Top + Stego->Height/2 - Open->Height/2;
     Open->ShowModal();
            int   SHAG   =    StrToInt(Open->Edit1->Text);     //
считываем ключ
```

Продолжение приложения А

```
      int KOL  =  StrToInt(Open->Edit2->Text);

      TPoint Next;    // позиция следующего кодируемого пиксела
```

```
      TPoint First;   // позиция кодированного пиксела
      BYTE TextByte;  // переменная для хранения символа

      char *result = new char[KOL+1];
      strset(result, '\0');
      int *BitMass = new int[8]; // массив для хранения битов

          for (int i = 0; i < KOL; i++)   // перебор символов
дешифруемого сообщения
               {
                for (int j = 0; j < 8; j++) // перебор битов
символа
                   {
                    Next = GetPosition(First, SHAG, WIDTH);

                    long    COLORR    =    GetPixel(Bmp->Canvas-
>Handle,First.x,First.y);  // получили цвет
                    BYTE R=GetRValue(COLORR);    //
                    //BYTE  G=GetGValue(COLORR);      // получили
каналы
                    //BYTE B=GetBValue(COLORR);    //

                    BitMass[j] = GetBitValue(R, 8);

                    First = Next;
                   }
                 result[i] =  BinaryToByte (BitMass);
                 for (int k = 0; k < 8; k++) BitMass[k] = 0;
               }

      RichEdit1->Clear();
      result[KOL] = '\0';
      RichEdit1->Text = result;
     }
     //---------------------------------------------------------------

     void __fastcall TStego::FormShow(TObject *Sender)
     {
     int   screenW   =   GetSystemMetrics(SM_CXSCREEN);//Получить
ширину экрана
     int   screenH   =   GetSystemMetrics(SM_CYSCREEN);//Получить
высоту экрана

     Stego->Left = screenW/2 - Stego->Width/2;
     Stego->Top =  screenH/2 - Stego->Height/2;
     }
```

Приложение Б

```
     #include <vcl.h>
     #include <fstream.h>
     #pragma hdrstop
```

```
#include "OpenKey.h"
#include "Steganografia.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TOpen *Open;
//---------------------------------------------------------------------------
__fastcall TOpen::TOpen(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TOpen::Button1Click(TObject *Sender)
{
Stego->OpenDialog1->Filter = "Key DaVinchi|*.DVKey";
Stego->OpenDialog1->Execute();
int SHAG; int KOL;

  if (Stego->OpenDialog1->FileName != "")
     {
  ifstream OPEN;
  OPEN.open(Stego->OpenDialog1->FileName.c_str());

  //             ------ШАГ--------             --КОЛ-ВО СЛОВ--
  OPEN>>SHAG>>KOL;
  Edit1->Text = IntToStr(SHAG);
  Edit2->Text = IntToStr(KOL);
     }

}
//---------------------------------------------------------------------------

void __fastcall TOpen::Button2Click(TObject *Sender)
{
  if ((Edit1->Text != "") && (Edit2->Text != ""))
     {
        int SHAG =  StrToInt(Edit1->Text);  // проверяем что
введены целые числа
        int KOL =  StrToInt(Edit2->Text);
        Open->Close();
     }

}
```

Приложение В

```
#include <vcl.h>
#include <fstream.h>
#pragma hdrstop

#include "Save.h"
#include "Steganografia.h"
//---------------------------------------------------------------------
-

#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
//---------------------------------------------------------------------
-

__fastcall TForm1::TForm1(TComponent* Owner)
     : TForm(Owner)
{
}
//---------------------------------------------------------------------
-

void __fastcall TForm1::Button1Click(TObject *Sender)
{

Stego->SaveDialog1->Execute();
if (Stego->SaveDialog1->FileName == "")
     {
        MessageBox(NULL, "Ключ не сохранен...", "Ошибка!",
MB_OK | MB_ICONERROR);
        return;
     }

  ofstream READ;
  READ.open(Stego->SaveDialog1->FileName.c_str());

  //             ------ШАГ-------             --КОЛ-ВО СЛОВ--
-
  READ<<StrToInt(Label9->Caption)<<endl;
  READ<<StrToInt(Label12->Caption);

Form1->Close();
}
```