МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ «ВИТЕБСКИЙ ГОСУДАРСТВЕННЫЙ

УНИВЕРСИТЕТ ИМЕНИ П.М. МАШЕРОВА»

Факультет\_\_\_\_\_\_\_\_Информационных технологий\_\_\_\_\_\_\_\_\_\_

(название факультета)

Кафедра\_\_\_\_\_\_\_Информатики и информационных технологий\_\_\_\_\_\_\_

(название кафедры, за которой закреплена дисциплина)

КУРСОВОЙ ПРОЕКТ

по дисциплине Технологии разработки информационных систем

(название дисциплины)

Технологии кодирования и декодирования

информации в ИС

(название темы работы заглавными буквами)

\_\_\_\_\_\_\_Овчаренко Валерий Иванович\_\_\_\_\_\_\_\_

(фамилия, имя, отчество студента (слушателя)

\_\_\_\_\_\_\_\_\_\_\_3 курс, 31з группа\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(курс, группа)

\_\_\_\_\_\_\_\_Шедько Василий Викторович\_\_\_\_\_\_\_\_

(фамилия, имя, отчество)

\_\_\_\_\_\_\_\_\_\_старший преподаватель\_\_\_\_\_\_\_\_\_\_\_\_

(должность, учёная степень, звание)

Витебск,2019 **Содержание**

[Введение 3](#_Toc10673881)

[1. История кодирования информации 5](#_Toc10673882)

[1.1. Развитие письменности 5](#_Toc10673883)

[1.2. Шифр Цезаря 6](#_Toc10673884)

[1.3. Азбука Морзе 8](#_Toc10673885)

[2. Кодирование информации в комьютере 11](#_Toc10673886)

[2.1. Преобразование сигнала из аналогового в цифровой 11](#_Toc10673887)

[2.2. Представление чисел в компьютере 12](#_Toc10673888)

[2.3. Кодирование символов 14](#_Toc10673889)

[2.4. Код Грея 15](#_Toc10673890)

[2.5. Кодирование информации 17](#_Toc10673891)

[3. Разработка программного продукта 19](#_Toc10673892)

[3.1. Выбор инструментальных средств разработки 19](#_Toc10673893)

[3.2. Алгоритмическое обеспечение проекта 19](#_Toc10673894)

[4. Пользовательский интерфейс программного продукта 21](#_Toc10673895)

[4.1. Главное окно программы 21](#_Toc10673896)

[4.2. Страница «Шифр Цезаря» 22](#_Toc10673897)

[4.3. Страница «Азбука Морзе» 24](#_Toc10673898)

[4.4. Страница «Числа» 27](#_Toc10673899)

[4.5. Страница «Строки» 29](#_Toc10673900)

[4.6. Страница «Цвета» 30](#_Toc10673901)

[4.7. Страница «Код Грея» 31](#_Toc10673902)

[4.8. Окно «О программе» 33](#_Toc10673903)

[Заключение 34](#_Toc10673904)

[Список использованных источников 35](#_Toc10673905)

Введение

Для передачи информации люди используют естественные языки. В повседневной жизни мы общаемся с помощью неформальной речи, а в деловой сфере используем формальный язык.

Сегодня для передачи и отображения информации мы используем вычислительную технику, которая «не понимает» наш язык без специальных операций – кодирования и декодирования.

Прежде чем разобраться с основами процедуры кодирования, следует ознакомиться с несколькими простейшими понятиями.

Код – это набор любых символов или других визуальных обозначений информации, который образует представление данных. В компьютерной технике под кодом подразумевают отдельную систему знаков, которые используют для обработки, передачи и хранения сообщений и файлов.

Кодирование – это процесс преобразования текстовой информации в код. Кодов существует огромное количество. Каждый из них отличается своим алгоритмом работы и алфавитом. К примеру, компьютер, смартфон, ноутбук и любые другие компьютерные устройства работают с двоичным кодом. Двоичный код использует алфавит, который состоит из двух символов – «0» и «1».

Декодирование – это процедура обратная к кодированию. Декодировщик обратно превращает код в понятную для человека форму представления данных. Среди известных примеров постоянной работы с декодированием можно отметить азбуку Морзе: для «прочтения» сообщения нужно сначала преобразовать полученный код в слова.

В компьютерной технике кодирование происходит, когда пользователь вводит любую информацию в систему – создает файлы, печатает текст и так далее. Для понимания обычных букв кириллицы или латиницы они превращаются в набор нолей и единиц. Чтобы отобразиться на экране компьютера, система проводит декодирование числовой последовательности и выводит результат на экран. Все эти действия выполняются за тысячные доли секунды.

**Цель курсовой работы**: Понять в каких случаях какие способы и алгоритмы кодирования и декодирования информации используются.

**Задача курсовой работы**: Изучить способы кодирования и декодирования информации в информационных системах.

**Объекты исследования:** Способы кодирования информации информационных системах.

[1, 4, 7, 11]

©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©

©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©

©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©

©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©

©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©

©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©

©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©

©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©

©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©© ©©©©©©©©©©©©©©©©©©©©©©©©©©

©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©

©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©

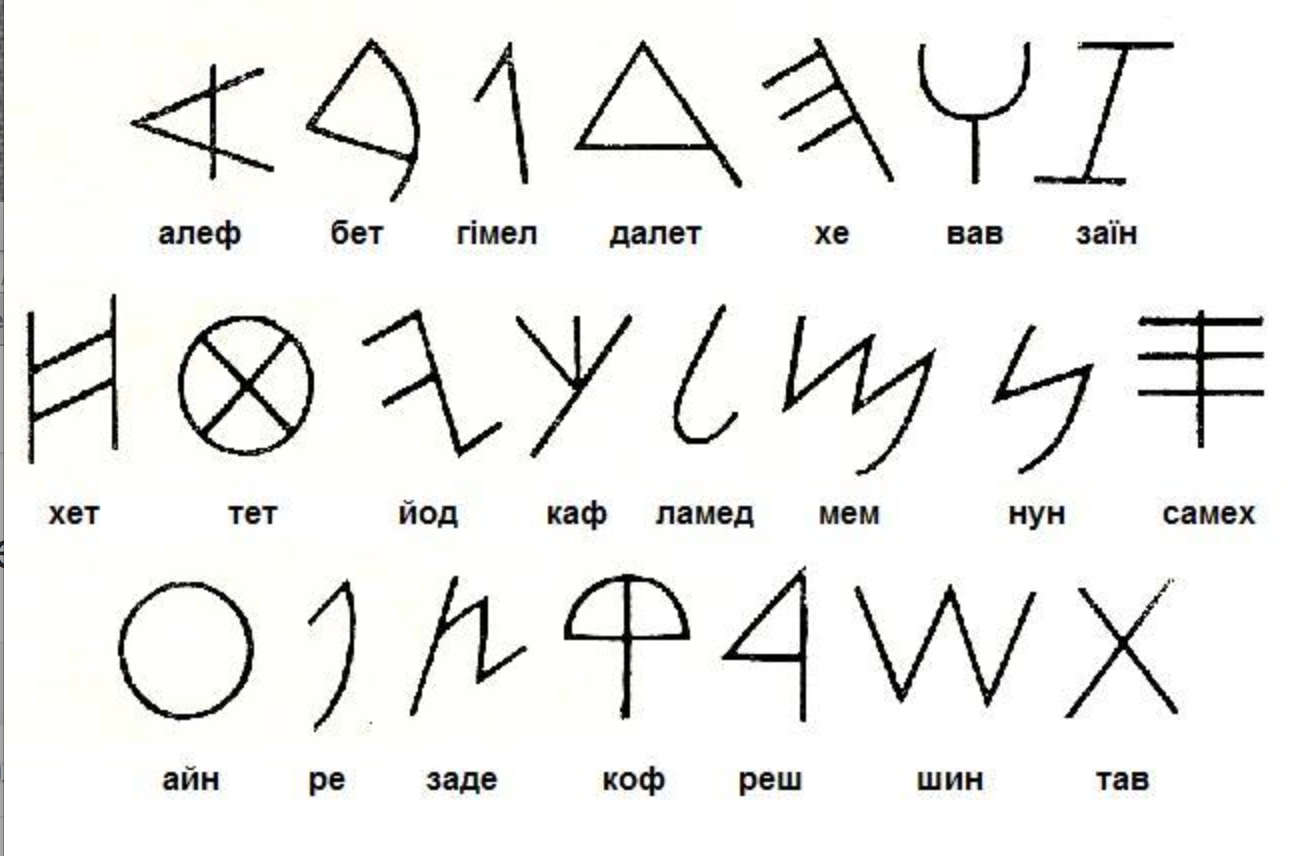
©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©

©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©

# История кодирования информации

## Развитие письменности

Прежде чем закодировать – преобразовать информацию из одного представления в другое, эту информацию требуется записать. Для того чтобы информацию как–то записать сначала люди придумали пиктограммы (Рисунок 1.1.1 – Египетские иероглифы (а)) – это рисунки обозначающие понятия из человеческого мира и быта.

**Рисунок 1.1.1 – Египетские иероглифы(а) и Финикийский алфавит (б)**

Рисовать такие пиктограммы было долго, и требовалось достаточное время на обучение этому, и нарисовать понятный рисунок получится не у каждого. Потом финикийцы придумали первый алфавит, в котором каждому звуку их речи соответствовала одна буква (Рисунок 1.1.1 – Финикийский алфавит (б)).

В истории любой страны всегда присутствуют войны, не только поэтому, но и по другим причинам рано или поздно определённую информацию нужно защитить от нежелательного прочтения, для этого люди изобретали разные способы зашифровать текст, записанный символами алфавита некоторого языка. А другие люди соответственно пытались этот шифр взломать.

В последствии это привело к появлению Криптографии – наука о методах обеспечения конфиденциальности (невозможности прочтения информации посторонним), целостности данных (невозможности незаметного изменения информации), аутентификации (проверки подлинности авторства или иных свойств объекта),

Одним из первых устройств для шифрования текста была Скитала – жезл, на который спиралью наматывалась лента из папируса, и вдоль жезла по этой ленте писались слова (Рисунок 1.1.2 – Скитала). Если взять жезл другого диаметра, то прочесть текст не получится.



**Рисунок 1.1.2 – Скитала**

## Шифр Цезаря

Шифр Цезаря, также известный как шифр сдвига, код Цезаря или сдвиг Цезаря — один из самых простых и наиболее широко известных методов шифрования.

Шифр Цезаря — это вид шифра подстановки, в котором каждый символ в открытом тексте заменяется символом, находящимся на некотором постоянном числе позиций левее или правее него в алфавите. Например, в шифре со сдвигом вправо на 3, буква «А» была бы заменена на букву «Г», и буква «Б» станет буквой «Д», и так далее.

Шифр назван в честь римского полководца Гая Юлия Цезаря, использовавшего его для секретной переписки со своими генералами.

Шаг шифрования, выполняемый шифром Цезаря, часто включается как часть более сложных схем, таких как шифр Виженера, и всё ещё имеет современное приложение в системе ROT13. Как и все моноалфавитные шифры, шифр Цезаря легко взламывается и не имеет почти никакого применения на практике.



**Рисунок 1.2.1 – Гай Юлий Цезарь**

Если сопоставить каждому символу алфавита его порядковый номер (нумеруя с 0), то шифрование и дешифрование можно выразить формулами модульной арифметики:

где x – символ открытого текста, а y – символ шифрованного текста, n – мощность алфавита, и k – ключ.

С точки зрения математики шифр Цезаря является частным случаем аффинного шифра.

Шифрование с использованием ключа k = 3. Буква «Е» «сдвигается» на три буквы вперёд и становится буквой «З». Твёрдый знак, перемещённый на три буквы вперёд, становится буквой «Э», буква «Я», перемещённая на три буквы вперёд, становится буквой «В», и так далее:

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

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

Оригинальный текст: «Съешь же ещё этих мягких французских булок, да выпей чаю».

Шифрованный текст получается путём замены каждой буквы оригинального текста соответствующей буквой шифрованного алфавита: «Фэзыя йз зьи ахлш пвёнлш чугрщцкфнлш дцосн, жг еютзм ъгб».

## Азбука Морзе

Телеграф Шаппа Первым техническим средством кодирования данных был созданный в 1792 году телеграф Шаппа. Устройство передавало оптическую информацию в простейшем виде с помощью специальной таблицы кодов, в которой каждой букве латинского алфавита соответствовала одна фигура.

В результате, телеграф мог отобразить и передать набор фигур. Скорость передачи таких сообщений составляла всего два слова в минуту. Технология такого обмена сообщениями была актуальна больше ста лет после создания телеграфа Шаппа.

Телеграф Морзе Созданный в 1837 году стал революционном устройством кодирования и декодирования информации. Принцип кодирования заключался в преобразовании любого сообщения в три символа алфавита:

Длинный сигнал – тире;

Короткий сигнал – точка;

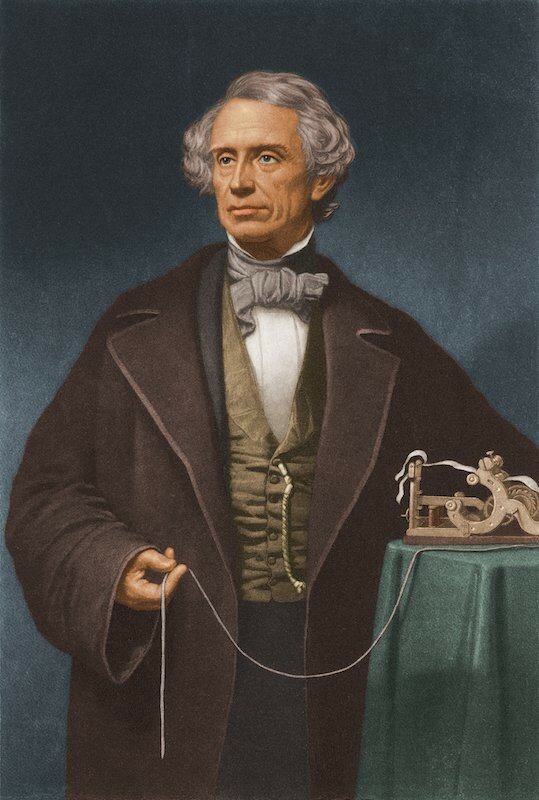
Нет сигнала – пауза.

Подобная связь используется по сей день в мореходной сфере для мгновенной передачи сообщений между суднами.

В 1899 году А. Попов создал первый в мире беспроводной телеграф или радиоприемник. Принцип его работы заключался в кодировании электрических сигналов азбукой Морзе и её дальнейшей передаче на длительные расстояния.

Позже был изобретен телеграф Бодо, который решал проблему неравномерности кода и сложность декодирования. Следующий этап в развитии кодирования – это создание вычислительных машин и их работа с бинарной системой исчисления.

Азбука Морзе («Морзянка» в народе), код Морзе – способ знакового кодирования, представление букв алфавита, цифр, знаков препинания и других символов последовательностью сигналов: длинных («тире») и коротких («точек»). За единицу времени принимается длительность одной точки. Длительность тире равна трём точкам. Пауза между элементами одного знака — одна точка, между знаками в слове — 3 точки, между словами — 7 точек. Назван в честь американского изобретателя и художника Сэмюэля Морзе.



**Рисунок 1.3.1 – Сэмюэль Морзе**

Буквенные коды (собственно «азбука») были добавлены коллегой Морзе, Альфредом Вейлем — факт, который Морзе впоследствии всячески отрицал (а заодно приписывал себе изобретение телеграфа как такового). Вейлем же, возможно, была придумана и цифровая часть кода. А в 1848 году код Вейля Морзе был усовершенствован немцем Фридрихом Герке (англ.). Код, усовершенствованный Герке, используется до настоящего времени.

[1,3,13,14]

Sdfskj23 scvhsiudfhksdjf8 sldmvnsknv3 skdhvkjchv7 lkdshglkajsdhf457 lskcvnskjdhfksjcvn8798 sdkjghkjh234 jdf87 ksjdth34hs87 kdmgvnhsjehfd7 lksdghasdlkfj4t7j9384759234 jgldk3

dvnkjns©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©© lskcvnskjdhfksjcvn8798 sdkjghkjh234 jdf87 ksjdth34hs87 kdmgvnhsjehfd7 lksdghasdlkfj4 ©©©©©©©©©©©©©©©©©©©©©©©©© jhsdioias3jdkghuhv lkcjvbuhwie3uriohg lskdjoisvhj3nwngvnsjchf oiwe6uvnkxjnclkvjslkdhfoiwhef woeihfsdh2vmcnvskdjfhgisdk, x mxcnvjhsu9erysjdhmncv. scvhlkfgnnxkc3hfuwyfo0jlkzn7vshvbjlkjf;

slcmvnskdjg234234©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©

# Кодирование информации в комьютере

## 2.1. Преобразование сигнала из аналогового в цифровой

Любой компьютер – это электронная техника, собранная из радиоэлементов, а радиоэлементы работают по принципам квантовой физики и могут принимать одно из двух состояний: включено (электрический ток проходит) и выключено – радиоэлемент закрыт.

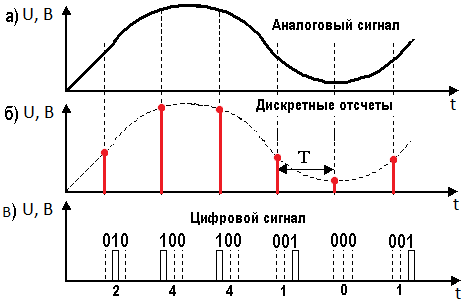
По этой причине и для увеличения скорости работы системы в компьютерах используют двоичную систему счисления для записи чисел. Система счисления – это способ записи чисел с помощью набора цифр, как слова записываются буквами из алфавита.

В природе все сигналы и электрический ток тоже имеют форму математического синуса. Электрический ток – это импульсы одного и того же напряжения, повторяющиеся через одинаковые промежутки времени (периоды). В компьютерах аналого-цифровое преобразование сигнала осуществляется микросхемой, расположенной на звуковой карте.

Для преобразования сигнала из аналогового вида в цифровой используется устройство под названием Аналого-цифровой преобразователь (сокращённо АЦП), а для обратного преобразования из цифрового сигнала в аналоговый используется соответствующее устройство Цифро-аналогового преобразователя (сокращённо ЦАП).

Аналого-цифровое преобразование соответствует математическому интегралу и осуществляется следующим способом:

* Сначала электроника много раз в секунду считывает напряжение электрических импульсов (частота дискретизации) – таким образом получается десятичное значение цифрового сигнала;
* Потом это десятичное значение переводится в соответствующее двоичное, где нулю соответствует напряжение до двух с половины вольт, а единице от двух с половиной до пяти вольт.



**Рисунок 2.1.1 – Аналого-цифровое преобразование**

## 2.2. Представление чисел в компьютере

Так как компьютер – это электрический прибор, собранный из радиоэлементов, а радиоэлементы могут находится в одном из двух состояний, то числа внутри компьютера хранятся в двоичном виде.

На ряду с обычной записью числа (десятичное число 235) существует экспоненциальная форма:

2.35Е+2

где после буквы «Е» указывается степень основания системы счисления.

Для представления двоичных чисел существуют следующие способы:

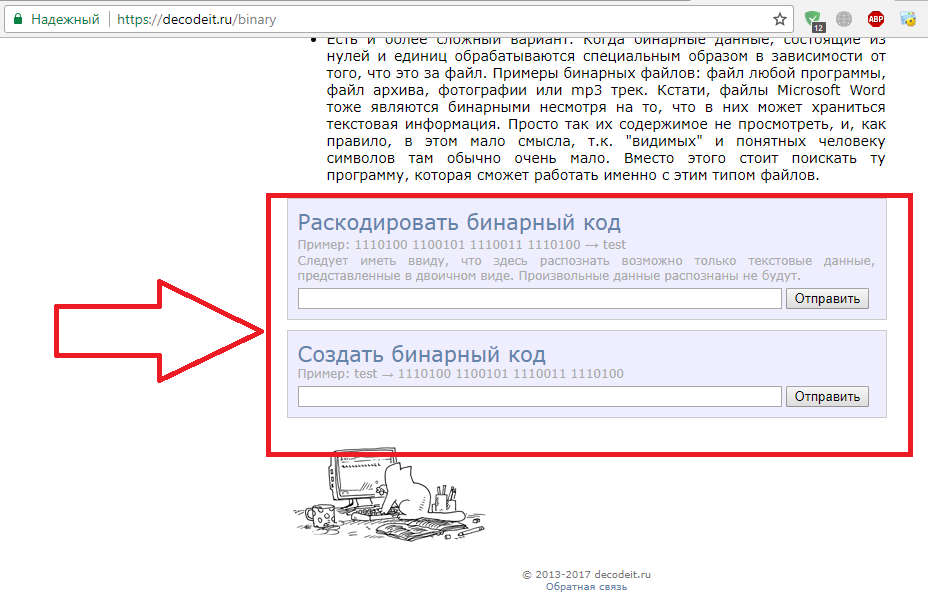
* прямой код, где для записи отрицательного числа первый бит является знаковым (0 – «+» и 1 – «-»);
* обратный код, где для записи отрицательных чисел разряды числа в прямом коде инвертируют (1 меняют на 0 и на оборот), для положительных чисел ничего не меняется;
* дополнительный код, где к числу в обратном коде прибавляем 1.

Дополнительный код позволяет заменить операцию вычитания на операцию сложения и сделать операции сложения и вычитания одинаковыми для знаковых и без знаковых чисел.

Ещё в компьютере используются системы следующие счисления:

* двоичная «11101011»;
* восьмеричная (вспомогательная) «535»;
* десятичная «235»;
* шестнадцатеричная «EB» используется для сокращённой записи чисел.

В повседневной жизни мы используем десятичную систему исчисления, алфавит которой представлен в виде чисел от 0 до 9. Онлайн-кодировщики для быстрого преобразования любого текста в набор символов бинарной или других систем исчисления удобнее использовать автоматические кодировщики. Также, они могут декодировать текст, самостоятельно определяя, какую систему использовал пользователь для кодировки. Популярным сервисом для создания или расшифровки двоичного кода является DecodeIT. Ресурс показывает высокую точность преобразования в обе стороны и отличается очень простым пользовательским интерфейсом.



**Рисунок 2.2.1 – Онлайн кодировщик DecoderIT**

## 2.3. Кодирование символов

Кодирование символов – это еще одна важная часть работы любого компьютерного устройства. От вышеописанных числовых систем она отличается тем, что кодирование происходит уже на этапе работы программы с определенным текстом, сообщением и другим видом данных. Для кодирования символов используются различные стандарты, среди которых Юникод, ASCII, UTF-8 и другие. Зачем нужна кодировка символов?

Любые символы на экране компьютера или смартфона отображаются за счет двух вещей:

1 Векторного представления;

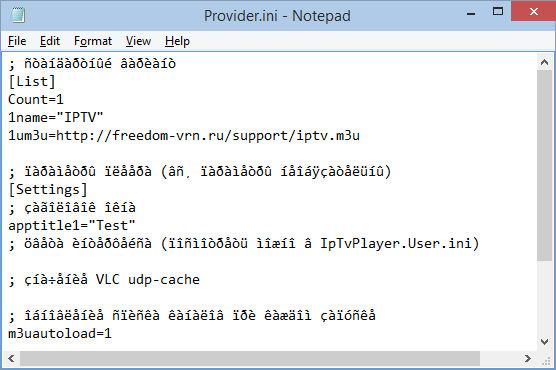
2 Предустановленных знаков и их кода.

Знаки – это шрифты, которые поддерживаются устройством. В ОС Windows они находятся в окне Панель управления (директория «Шрифты»). С помощью этой папки вы можете добавлять или удалять существующие представления символов. С помощью программного кода выбирается нужное векторное направление символа и его изображение из папки «Шрифты».

Таким образом, на экране появляется буква и текст. За установку шрифтов отвечает операционная система вашего компьютера, а за кодировку текста – программы, в которых вы набираете или просматриваете текстовые данные. Любая программа, к примеру стандартный Блокнот, в процессе открытия считывает кодировку каждого знака, производит декодирование данных и выводит информацию для просмотра или дальнейшего редактирования пользователем.

Разбирая код, приложение обрабатывает кодировку знака и ищет его соответствие в поддерживаемом для этого же документа шрифте. Если соответствие не найдено, вместо текста вы увидите набор непонятных символов.

Чтобы символы кириллицы и латиницы открывались без проблем в большинстве программ, было предложено ввести стандарты кодирования. Один из наиболее популярных – это Юникод (или Unicode). Он поддерживается практически всеми существующими шрифтами и программным обеспечением. Также, широко используются технологии UTF-8, ASCII.



**Рисунок 2.3.1 – Программа Notepad**

Если в программе текст отображается в нечитабельной форме, пользователь может самостоятельно его декодировать. Для этого достаточно зайти в настройки текстового редактора и сохранить файл с кодировкой Юникод или другими популярными форматами кодирования. Затем откройте файл заново, текст должен отображаться в нормальном режиме.

## 2.4. Код Грея

Код Грея – двоичный код, иначе зеркальный код, он же код с отражением, в котором две «соседние» (в упорядоченном, то есть лексикографическом, наборе) кодовые комбинации различаются только цифрой в одном двоичном разряде. Иными словами, расстояние Хэмминга между соседними кодовыми комбинациями равно 1.

Наиболее часто на практике применяется рефлексивный двоичный код Грея, хотя в общем случае существует бесконечное множество кодов Грея со значениями цифр в разрядах, взятых из различных алфавитов. В большинстве случаев, под термином «код Грея» понимают именно рефлексивный бинарный код Грея.

Код назван в честь исследователя Фрэнка Грея, работавшего в лаборатории «Bell labs». Грей запатентовал (патент № 2632058) и впервые использовал этот код в своей импульсной системе связи.



**Рисунок 2.4.1 – Фрэнк Грей**

Код Грея используется в передаче меняющихся цифровых сигналов в отсутствие тактового сигнала синхронизации (например, во многих видах датчиков). Представим себе, что код (обычный двоичный) перескакивает 3→4, или 0112 → 1002. Если из-за несовершенства считывателя мы прочитаем первый бит от 011, а остальные два — от 100, мы получим 0002=0 — число, далёкое от реальных значений. В коде Грея никаких посторонних значений не будет: перескок будет в одном разряде, 010G → 110G, и мы считаем либо старое 010G=3, либо новое 110G=4.

## 2.5. Кодирование информации

Часто возникает необходимость не только закодировать информацию, но и скрыть её содержимое от посторонних. Для таких целей используется шифрование. Простыми словами, шифрование – это кодирование информации, но не с целью её корректного представления на экране компьютера, а с целью сокрытия данных от тех, кому не положено получать доступ к шифрованной информации. Алфавит шифрования состоит из двух элементов:

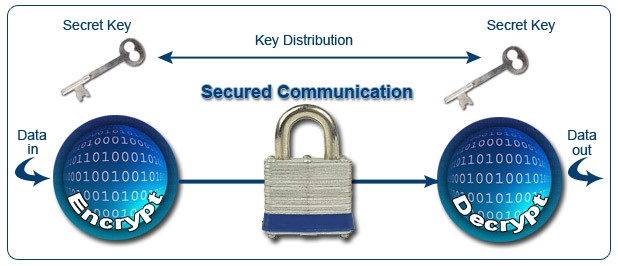
Алгоритм – уникальная последовательность математических действий с двоичными числами;

Ключ – бинарная последовательность, которая добавляется к шифруемому сообщению.

Дешифрование – это обратный процесс к защитному кодированию, который подразумевает превращение данных в первоначальный вид с помощью известного ключа. Криптография – это наука о шифровании данных. Всего различают два раздела криптографии:

Симметричная – в таких криптосистемах кодирования для шифрования и дешифрования используют один и тот же ключ. Недостаток системы – низкая стойкость ко взлому;

Ассиметричная – для шифрования используются закрытый и открытый ключ.



**Рисунок 2.5.1 – Принцип работы защищённого соединения**

Таким образом, посторонний человек не сможет расшифровать (декодировать) сообщение, даже если алгоритм известен. Где используется криптография? Кодирование информации с целью шифрования используется уже более трех тысяч лет. Истории известны первые попытки шифрованной передачи сообщений между известными полководцами царями и просто высокопоставленными людьми.

Сегодня без криптографии невозможно существование всей банковской системы, ведь каждая карта, каждая авторизация в онлайн-банкинге требует наличия защищенного соединения, при котором злоумышленник не сможет похитить ваши деньги или подобрать пароль.

Также, шифрованное кодирование используется в обычных социальных сетях, мессенджерах. К примеру, Telegram – мессенджер, главной особенностью которого является кодирование сообщений пользователей таким образом, чтобы никто посторонний не смог взломать переписку. Также, алгоритмы шифрования встроены во все операционные системы, облачные хранилища. Они нужны для защиты ваших личных данных.

[2, 5, 8, 10]

# Разработка программного продукта

## 3.1. Выбор инструментальных средств разработки

Пользовательское программное обеспечение «WeatherNetwork» будет разработано в среде программирования Microsoft Visual Studio 2019 на объектно–ориентированном языке высокого уровня Си Шарп.

«Microsoft Visual Studio Ultimate 2019». Визуальное программирование позволяет быстро создавать интерфейс программы и делать его более качественным за счёт наилучшего расположения информации окна на экране монитора, избежать многих ошибок уже на этапе проектирования.

Интегрированная среда разработки Visual Studio – полнофункциональная интегрированная среда разработки для Windows и веб–приложений и облачных приложений.

Visual Studio позволяет быстро и эффективно писать код, не теряя из виду контекст текущего файла. Можно легко углубиться в подробности, такие как структура вызова, связанные функции, возвраты и состояние тестирования. Также доступны рефакторинг кода, нахождение и устранение ошибок в коде.

## 3.2. Алгоритмическое обеспечение проекта

В ходе проектирования программного продукта проведена следующая работа:

− исследована предметная область, собрана необходимая информация (текстовая, графическая и др.);

− определены цели и задачи проектирования приложения;

− выбран язык программирования.

При разработке структуры проекта необходимо:

− спланировать структуру приложения (разделы, навигация, программные функции);

− разработать дизайн приложения;

Интерфейс программы должен быть простым и удобным для пользователя. При этом программный продукт должен хранить необходимую информацию и выполнять заявленные функции. Необходимо также продумать удобную навигацию по приложению.

Для реализации программного продукта выбран язык программирования С#.

[7,8, 9, 11]

©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©©l89k8797 we234

# Пользовательский интерфейс программного продукта

## 4.1. Главное окно программы

Программа называется «КИД Инфо» – это сокращение от «Кодирование и декодирование информации».

Главное окно программы представляет собой основное рабочее пространство для пользователя, програмно представлено объектом MainForm, наследованным от стандартного класса Form. Остальные окна программы представляют собой аналогичные объекты.

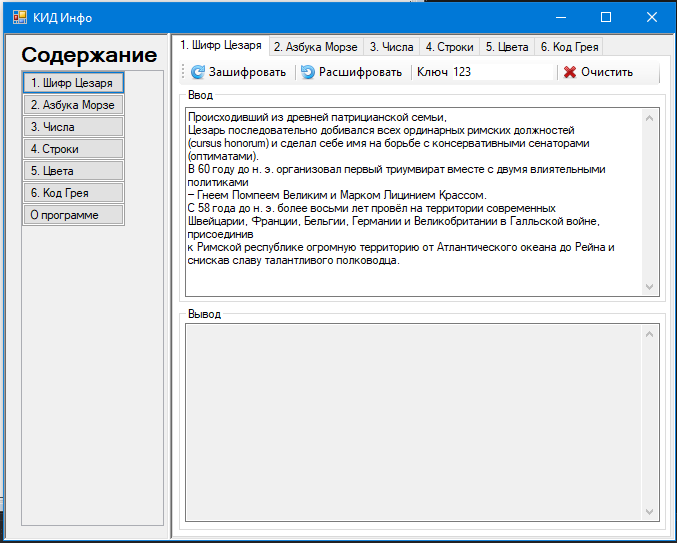


Рисунок 4.1.1 – Главное окно программы

На главной форме размещён компонент splitContainer, который предназначен для разделения формы на две динамических панели, которые нужны для упорядочивания других элементов управления.

Левая панель – Panel1 компонента splitContainer содержит компонент labelTitle для отображения надписи «Содержание», и компонент contentDataGrid для отображения кнопок самого содержания.

Для отображения кнопок правой панели используется следующий код:

Листинг 4.1.1 – Обработчик события Shown Главной формы

private void MainForm\_Shown ( object sender, EventArgs e ) {

tabControl.SelectedIndex = 0;

foreach ( TabPage page in tabControl.TabPages ) {

contentDataGrid.Rows.Add( page.Text );

}

contentDataGrid.Rows.Add( "О программе" );

}

Правая панель – Panel2 компонента splitContainer содержит компонент tabControl для отображения вкладок, на которых размещены страницы разделов содержания.

На каждой вкладке размещён соответствующий визуальный компонент, наследуемый от класса UserControl, и предназначен для отображения соответствующих содержимого страницы раздела и элементов управления на ней:

Для перехода на выбранный раздел программы при щелчке по соответствующей кнопке выполнится такой код:

Листинг 4.1.2 – Обработчик события CellClick компонента ContentDataGrid

private void ContentDataGrid\_CellClick ( object sender, DataGridViewCellEventArgs e ) {

if ( e.RowIndex < tabControl.TabPages.Count ) {

tabControl.SelectedIndex = e.RowIndex;

} else {

new AboutForm( ).ShowDialog( );

}

}

## 4.2. Страница «Шифр Цезаря»

Содержимое этой страницы представлено пользовательским визуального компонентом CaesarCrypt, наследованным от стандартного класса UserControl, и содержит следующие графические компоненты:

* textInputBox – поле для ввода исходного текста для шифровки и расшифровки, расположенное внутри визуального элемента управления groupBoxInput;
* textOutputBox – поле для ввода текста после шифровки и расшифровки, расположенное внутри визуального элемента управления groupBoxOutput;
* toolTripCC – панель для размещения элементов управления.

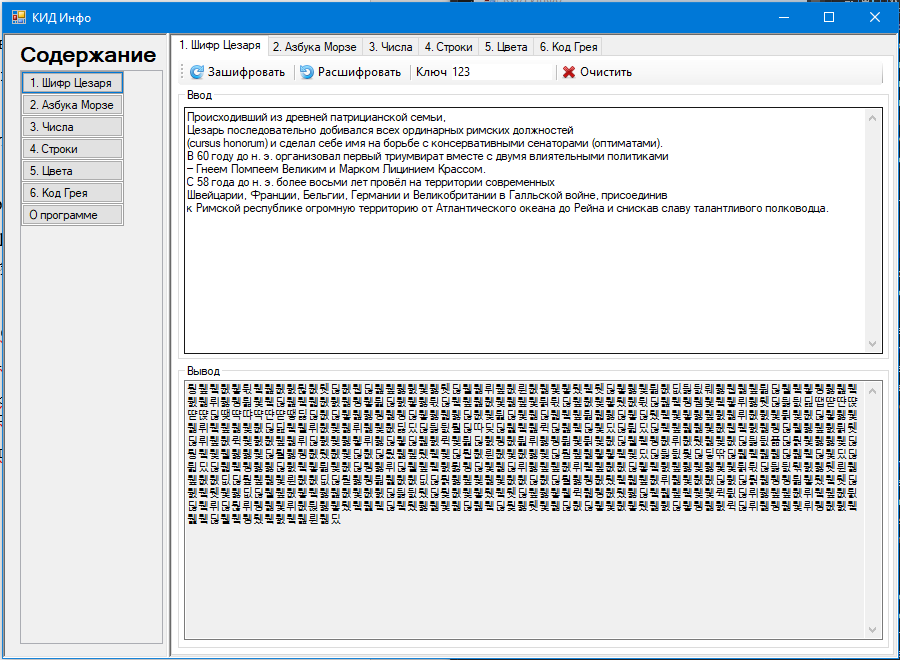


Рисунок 4.2.1 – Страница «Шифр Цезаря»

Визуальный элемент управления toolTripCC, предназначенный для отображения элементов управления, содержит следующие визуальные элементы управления:

* ButtonCrypt – это кнопка для выполнения операции шифрования способом Цезаря;
* ButtonEncrypt – это кнопка для выполнения операции дешифрования по способу Цезаря;
* keyLabel – визуальный элемент управления, используемый для отображения текста;
* textBoxKey – визуальный элемент управления, используемый для ввода текста (ключа для шифрования и дешифрования способом Цезаря);
* ButtonClear – это кнопка для очистки полей textInputBox и textOutputBox;
* Три визуальных компонента класса ToolStripSeparator для разделения элементов управления, размещённых на панели toolTripCC.

Для кодирования текста методом Цезаря после нажатия кнопки ButtonCrypt выполнится следующий код:

Листинг 4.1.1 – Обработчик события Click компонента ButtonCrypt

private void ButtonCrypt\_Click ( object sender, EventArgs e ) {

textBoxOutput.Clear( );

var key = textBoxKey.Text.GetHashCode( );

foreach ( int symbol in textBoxInput.Text ) {

textBoxOutput.Text += ( char ) ( ( symbol + key ) % UTF\_TABLE\_LENGTH );

}

}

Для декодирования текста по средствам метода Цезаря после нажатия кнопки ButtonEncrypt выполнится следующий код:

Листинг 4.2.2 – Обработчик события Click компонента ButtonEncrypt

private void ButtoncEncrypt\_Click ( object sender, EventArgs e ) {

textBoxOutput.Clear( );

var key = textBoxKey.Text.GetHashCode( );

foreach ( int symbol in textBoxInput.Text ) {

textBoxOutput.Text += ( char ) ( ( symbol - key ) % UTF\_TABLE\_LENGTH );

}

}

## 4.3. Страница «Азбука Морзе»

Содержимое этой страницы представлено пользовательским визуального компонентом CodeMorze, наследованным от стандартного класса UserControl, и содержит следующие графические компоненты:

* textInputBox – поле для ввода исходного текста для шифровки и расшифровки, расположенное внутри визуального элемента управления groupBoxInput;
* textOutputBox – поле для ввода текста после шифровки и расшифровки, расположенное внутри визуального элемента управления groupBoxOutput;
* toolTripCC – панель для размещения элементов управления.

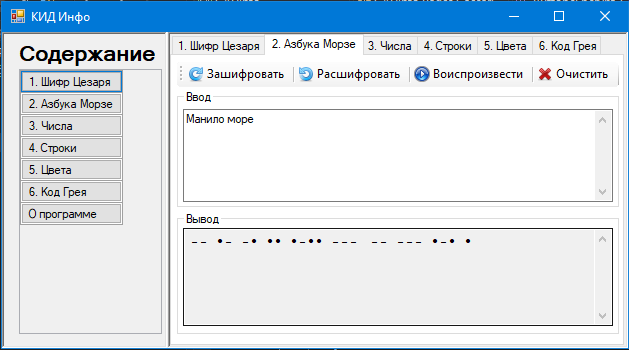


Рисунок 4.3.1 – Страница «Азбука Морзе»

Визуальный элемент управления toolTripCC, предназначенный для отображения элементов управления, содержит следующие визуальные элементы управления:

* ButtonCrypt – это кнопка для выполнения операции кодирования в код Морзе;
* ButtonEncrypt – это кнопка для выполнения операции декодирования из кода Морзе;
* ButtonClear – это кнопка для очистки полей textInputBox и textOutputBox;
* ButtonPlay – это кнопка для воспроизведения полученного кода Морзе;
* Три визуальных компонента класса ToolStripSeparator для разделения элементов управления, размещённых на панели toolTripCC.

Для кодирования текста в код Морзу после нажатия кнопки ButtonCrypt выполнится следующий код:

Листинг 4.3.1 – Обработчик события Click компонента ButtonCrypt

private void ButtonCrypt\_Click ( object sender, EventArgs e ) {

textBoxOutput.Clear( );

foreach ( var symbol in textBoxInput.Text ) {

textBoxOutput.Text += \_coder.GetMorze( symbol.ToString( ) );

}

}

Для декодирования в текст кода Морзе после нажатия кнопки ButtonEncrypt выполнится следующий код:

Листинг 4.3.2 – Обработчик события Click компонента ButtonEncrypt

private void ButtoncEncrypt\_Click ( object sender, EventArgs e ) {

textBoxOutput.Clear( );

var dot = '•';

var dash = '‒';

var word = string.Empty;

foreach ( var symbol in textBoxInput.Text ) {

if ( symbol == dot || symbol == dash ) {

word += symbol;

} else {

textBoxOutput.Text += word == string.Empty ? symbol.ToString( ) : \_coder.GetSymbol( word );

word = string.Empty;

}

}

}

Для воспроизведения полученного кода Морзе после нажатия кнопки ButtonPlay выполнится следующий код:

Листинг 4.3.3 – Обработчик события Click компонента ButtonPlay

private void ButtonPlay\_Click ( object sender, EventArgs e ) {

var dotTime = 30;

var dashTime = dotTime \* 3;

var spaceTime = dotTime \* 7;

var dot = '•';

var dash = '‒';

var frequency = 3000;

foreach ( var symbol in textBoxOutput.Text ) {

if ( symbol == dot ) {

Beep( frequency, dotTime );

Thread.Sleep( dashTime );

} else if ( symbol == dash ) {

Beep( frequency, dashTime );

Thread.Sleep( dashTime );

} else {

Thread.Sleep( spaceTime );

}

}

}

## 4.4. Страница «Числа»

Содержимое этой страницы представлено пользовательским визуального компонентом Numbers, наследованным от стандартного класса UserControl, и содержит следующие графические компоненты:

* groupBoxDecNumber – это визуальный компонент, представляющий собой контейнер с рамкой для организации пользовательского интерфейса, и предназначен для организации показа введённого числа в различных формах и системах счисления;
* groupBoxBinNumber – это визуальный компонент, представляющий собой контейнер с рамкой для организации пользовательского интерфейса, и предназначен для организации показа введённого двоичного числа в различных способах представления.

Визуальный элемент управления groupBoxDecNumber содержит следующий компоненты:

* textBoxDec – визуальный элемент управления для ввода текста (числа в десятичной системе счисления);
* buttonDecCalc – для преобразования введённого в textBoxDec числа и вывода его в различных системах счисления в соответствующие компоненты класса Label;
* объекты класса Label, предназначенные для вывода текста и введённого числа в различных формах и системах счисления.

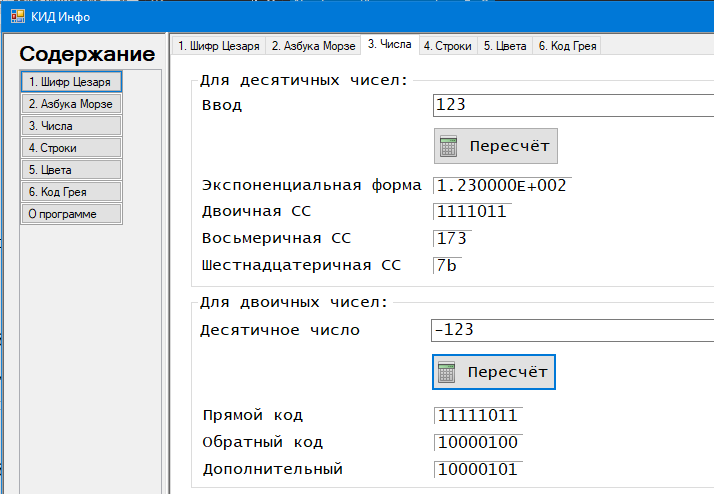


Рисунок 4.4.1 – Страница «Числа»

Визуальный элемент управления groupBinNumber содержит следующий компоненты:

* textBoxSignDec – визуальный элемент управления для ввода текста (числа в десятичной системе счисления);
* buttonBinCalc – для преобразования введённого в textBoxSignDec числа и вывода его в различных способах представления двоичных чисел в соответствующие компоненты класса Label;
* объекты класса Label, предназначенные для вывода текста и введённого числа в различных формах и системах счисления.

## 4.5. Страница «Строки»

Содержимое этой страницы представлено пользовательским визуального компонентом Strings, наследованным от стандартного класса UserControl, и содержит следующие графические компоненты:

* textBoxString – это визуальный компонент, представляющий собой поле для вводе текста или кодов символов через пробел;
* buttonConvertToCode – кнопка для преобразования текста введённого в textBoxString и вычисления хешей;
* buttonConvertToSymbols – кнопка для преобразования кодов символов введённых через пробел в textBoxString и в текст;
* labelOutput – это визуальный компонент для вывода кодов символов или текста;
* labelMyCode – это визуальный элемент управления для вывода результатов своей хеш-функции;
* labelStandatrCode – это визуальный элемент управления для вывода результатов стандартной хеш-функции.

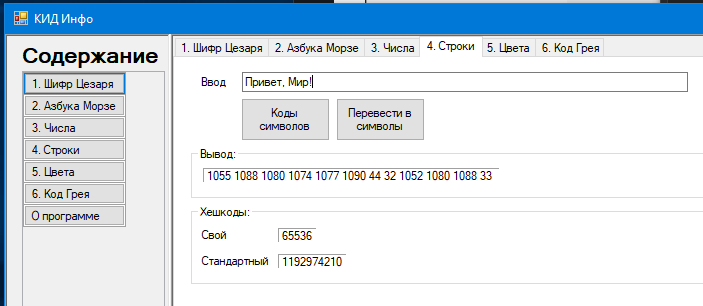


Рисунок 4.5.1 – Страница «Строки»

Для вычисления хеш-значения стоки была написана своя хеш-функция:

Листинг 4.5.1 – Своя хеш-функция

private long HashCode ( string text ) {

const int UTF\_TABLE\_LENGTH = 65\_536;

long hash = 1;

for ( int i = 0; i < text.Length; i++ ) {

hash = UTF\_TABLE\_LENGTH << ( ( ( ( int ) text[ i ] ) ) \* ( i + 1 ) );

}

return hash;

}

## 4.6. Страница «Цвета»

Содержимое этой страницы представлено пользовательским визуального компонентом Colors, наследованным от стандартного класса UserControl, и содержит следующие графические компоненты:

* pictureBoxImage – это визуальный компонент, предназначенный для вывода изображения;
* panelSelectColor – это визуальный элемент пользовательского интерфейса, предназначенный для показа цвета выбранного пиксела из pictureBoxImage;
* визуальные компоненты класса Panel (panelColorR, panelColorB и panelColorG), предназначены для отображения цвета соответствующих цветовых составляющих для цвета из пространства RGB;
* визуальные компоненты класса Label для вывода текстовой информации, отображают информацию о соответствующих составляющих цвета, в конкретном пространстве цветов;
* визуальные компоненты класса GroupBox, размещённые на данном пользовательском компоненте Colors, предназначены для группировки визуальных элементов пользовательского интерфейса в логически общие группы для удобства пользователя.

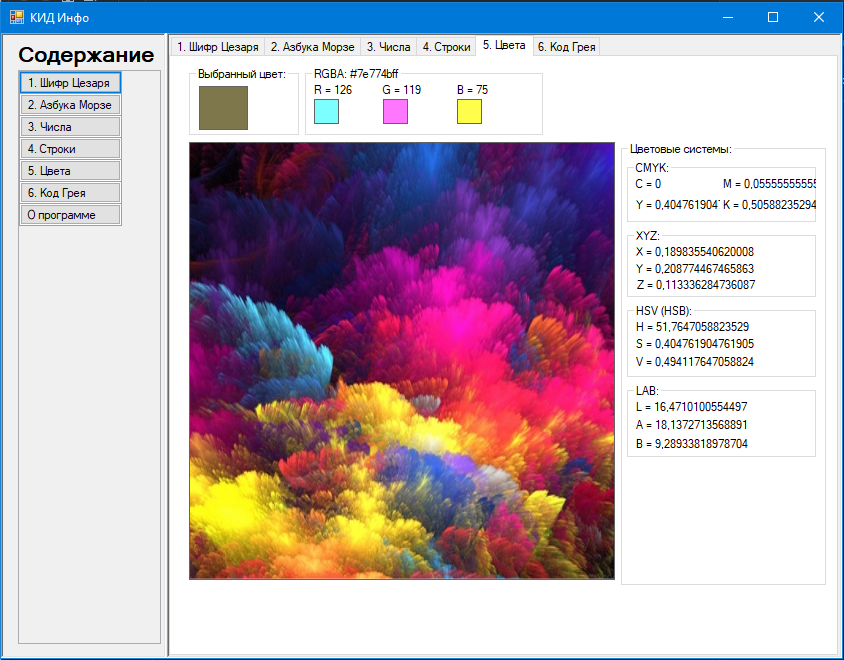


Рисунок 4.6.1 – Страница «Цвета»

## 4.7. Страница «Код Грея»

Содержимое этой страницы представлено пользовательским визуального компонентом Colors, наследованным от стандартного класса UserControl, и содержит следующие графические компоненты:

* textBoxInputGray – это визуальный компонент, предназначенный для вводе текста, двоичного числа в данном случае;
* визуальные компоненты класса Label для вывода текстовой информации, отображают информацию о введённом числе, преобразованном в десятичное число и соответствующие представления того же числа, но только в коде Грея;
* buttonConvertToGray – это кнопка предназначенная для преобразования введённого в textBoxInputGray двоичного числа и вывода преобразованного результата в коде Грея;
* buttonConvertOfGray – это кнопка предназначенная для преобразования введённого в textBoxInputGray двоичного числа в коде грея и вывода преобразованного результата в коде нормальном представлении;
* визуальные компоненты класса GroupBox, размещённые на данном пользовательском компоненте Colors, предназначены для группировки визуальных элементов пользовательского интерфейса в логически общие группы для удобства пользователя.

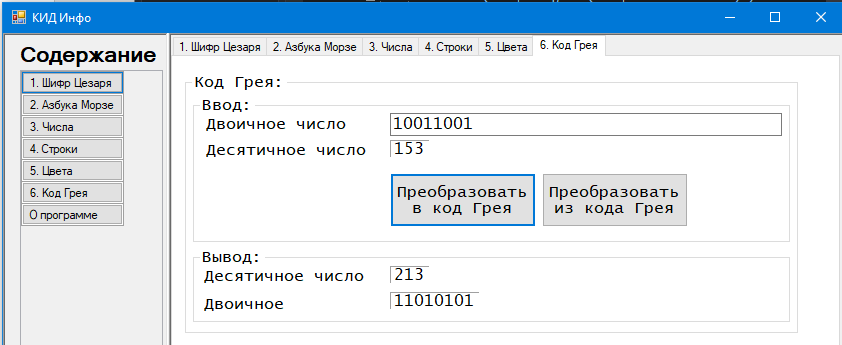


Рисунок 4.7.1 – Страница «Код Грея»

Для преобразования двоичного числа, введённого в textBoxInputGray в соответствующее число в коде Грея был написан следующий код:

Листинг 4.7.1 – Функция для преобразования числа в код Грея

string BinToGray(string number) {

isParse = int.TryParse(number, out var binNumber);

if (!isParse) {

MessageBox.Show("Ввели не правильное число ");

return string.Empty;

}

binNumber = Convert.ToInt32(number, 2);

return Convert.ToString(binNumber ^ (binNumber >> 1));

}

Для преобразования двоичного числа в коде Грея, введённого в textBoxInputGray в коде обычный вид написан следующий код:

Листинг 4.7.2 – Функция для преобразования числа в код Грея

string GrayToBin(string number) {

isParse = int.TryParse(number, out var binNumber);

if (!isParse) {

MessageBox.Show("Ввели не правильное число ");

return string.Empty;

} else {

binNumber = Convert.ToInt32(number, 2);

int sum;

for (sum = 0; binNumber > 0; binNumber >>= 1) {

sum ^= binNumber;

}

return Convert.ToString(sum);

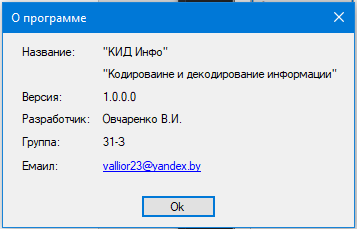
}

}

## 4.8. Окно «О программе»

Внешний вид окна «О программе» можно увидеть ниже. Внутри окошка размещены три компонента класса Label для отображения текстовой информации, и компонент класса.

Ещѐ на форме размещены компонент класса LinkLabel, который представляет собой текстовый элемент управления с адресом электронной почты для отправки сообщения разработчику. Таким образом пользователь при возникновении вопросов может обратиться к разработчику.



[4,6,8,15]

# Заключение

На данном этапе развития информационных технологий данная программа может упростить пользователям работу с помощью графического интерфейса, который не требуют большего изменения.

При выполнении курсового проекта были пройдены все этапы разработки специализированного прикладного программного обеспечения:

* изучение предметной области;
* изучения алгоритмов кодирования и декодирования информации;
* проектирование пользовательского интерфейса.

Кодирование информации используется сотни лет для удобной передачи данных между устройствами. С развитием технологий и переносом банковской сферы в техническую среду появилась необходимость в использовании алгоритмов кодирования, которые бы шифровали информацию, сохраняя её от несанкционированного доступа. Сегодня без технологий кодирования данных невозможна работа ни одного компьютера, смартфона, сайта или банковского счета.

В разных случаях и предметных областях нужны разные способы кодирования и декодирования информации и для разных случаев существуют свои алгоритмы, со временем появляются новые способы и алгоритмы кодирования информации.

В результате выполнения курсового проекта был разработан программный продукт, который можно использовать для обучения учащихся в качестве наглядного материала.

В ходе разработки курсового проекта использовались встроенный средства Visual studio для совместной разработки – Git, вот ссылка на размещённый репазиторий: https://github.com/Val2302/CADInfo.

[10,11,12]

Список использованных источников

1. Андреев А.М. Среда и хранилище: ООБД. – М.: «Мир ПК», 1997.

2. Бен Ватсон. C# 4.0 на примерах. Санкт–Петербург. : БХВ–Петербург, 2011, 590 с.

3. Буч Г. Объектно–ориентированный анализ и проектирование. 2–ое издание. – М.: «Бином», 1997.

4. Грабалов П.К. Компьютерная графика и основные графические редакторы. – Калининград,2003

5. Гуриков С. Р. Введение в программирование на языке Visual C#; Форум, Инфра–М, 2013. – 448 с.

6. Дейтел, Х. М.; Дейтел, П. Дж. Как программировать на С#; М.: Бином; Издание 4–е, 2005. – 390 с.

7. Джейсон, Прайс; Майк, Гандэрлой Visual C# .NET. Полное руководство; КОРОНА принт, 2004. – 960 c.

8. Лабор В.В. – С# Создание приложений для Windows (2003)

9. Мартин Р. С., Мартин М. Принципы, паттерны и методики гибкой разработки на языке С#; Символ–Плюс, 2011. – 768 с.

10. Нейгел, К. С# 2011 для профессионалов; Вильямс, 2012. – 763 с.

11. Пугачев С., Шериев А., Кичинский К. Разработка приложений для Windows 8 на языке C#; БХВ–Петербург, 2013. – 416 с.

12. Рихтер, Джефри CLR via C#. Программирование на платформе Microsoft .NET Framework 2.0 на языке C#; Питер, 2007. – 656 c.

13. Робинсон, С.; Корнес, О.; Глинн, Д. и др. C# для профессионалов; М.: Лори, 2005. – 396 c.

14. Фленов Михаил Библия С#; БХВ–Петербург, 2009. – 560 с.

15. А. Хейлсберг, М. Торгерсен, С. Вилтамут, П. Голд Язык программирования C#. Классика Computers Science. 4-е изд; 2011. – 784 с.