Муниципальное автономное образовательное учреждение дополнительного образования города Иркутска «Дворец детского и юношеского творчества»

Фестиваль-конкурс проектов «Становимся мастерами»

Учебный проект

«Приложение «Кодировщик Валера»»

Разработчик проекта:
Перелыгин Фёдор Алексеевич, 15 лет,
МАОУ ДО г. Иркутска «Дворец творчества»

Руководитель проекта:
Тимофеев Алексей Сергеевич,
педагог дополнительного образования
МАОУ ДО г. Иркутска «Дворец творчества»

Иркутск, 2024

Информационная карта проекта

Название проекта	«Приложение «Кодировщик Валера»»
Вид проекта	Учебный, Практико-ориентированный проект, среднесрочный (Январь-апрель 2024)
Автор проекта	Перелыгин Фёдор Алексеевич
Участники проекта *(при необходимости)*	Перелыгин Фёдор Алексеевич
Руководитель проекта	Тимофеев Алексей Сергеевич, педагог дополнительного образования, высшей квалификационной категории
Куратор проекта *(при необходимости)*	Тимофеев Алексей Сергеевич, педагог дополнительного образования, высшей квалификационной категории
Площадка реализации проекта	МАОУ ДО г. Иркутска Дворец творчества
Актуальность проекта	В мире есть люди, которые не хотят, чтобы кто-то посторонний читал их личные сообщения. По опросу учащихся класса 9м: 80% учащихся (24 человека из 30) хотят, чтобы их сообщения оставались в секрете. Также в данное время военные передают информацию друг другу по рации или через письма, что является не очень надёжным способом.
Проблема проекта	Секретность информации передаваемую через приложения, по телефону, в письмах или сообщениях
Гипотеза проекта *(при необходимости)*	Если разработать приложение, которое будет кодировать информацию, это позволит защитить персональные данные и переписку между собеседниками
Цель проекта	Создать функциональное и удобное приложение-кодировщик для шифрования информации.
Задачи проекта	1. Изучить существующие методы и алгоритмы шифрования для определения наиболее подходящего подхода к разработке приложения. 2. Создать пользовательский интерфейс приложения, обладающий простотой использования и интуитивно понятной навигацией. 3. Реализовать функционал шифрования и дешифрования данных в приложении, используя выбранные алгоритмы и методы. 4. Протестировать приложение, проведя тестирование функций шифрования и дешифрования.
Этапы реализации проекта	**Организованный(подготовительный)-21.11-23.01.2024:** Анализ проблемы, определение задач и конечного продукта. **Деятельностный – 24.01-03.02.2024:** Изучение литературы 03.02 Создание примерного кода 04.02-10.02 Доработка кода 11.02-23.02 Воплощение кода в приложение 23.02-01.03 Тестирование 01.03-03.03 **Заключительный – 04.03-29.03.2024**

	Поиск проблем в коде 04.03-07.03 Решение проблемы 08.03-12.03 Создание презентации 12.03-17.03 Оформление проекта 18.03-29.03
Планируемые результаты, продукт проекта	Приложение кодировщик
Методы достижения	Изучение литературы Анализ и синтез информации Практическое моделирование Эксперимент
Практическая значимость проекта	Люди могут не бояться, что их сообщения кто-то поймет, ведь человек сам может поменять в коде как будет кодироваться его сообщение.

Пояснительная записка

В предложенном проекте "Приложение кодировщик Валера" лежит основа кодирования информации, защиты данных между собеседниками, что в нынешнее время является важным для человечества. Автором был предложен свой способ кодирования и декодирования текста, а также разработано удобное портативное приложение для ПК, интерфейс которого будет понятен каждому. Было проведено многочисленное тестирование приложения, кодировка и декодировка текста, ручная кодировка, а также демонстрация приложения среди пользователей

Календарно-тематический план реализации творческого проекта «Приложение кодировщик Валера»

№	Мероприятие	Дата
1	анализ проблемы;	Январь 2023
2	Определение цели и задач;	Январь 2023
3	разработка плана мероприятий по реализации проекта.	Январь 2023
4	изучение литературы по теме проекта	Февраль 2023
5	написание примерного кода	Февраль 2023
6	тестирование, поиск ошибок	Март 2023
7	доработка кода с учетом ошибок	Март 2023
8	написания внешнего интерфейса GUI	Март 2023
9	тестирование интерфейса	Март 2023
10	проверка кодирования приложения	Март 2023
11	тестирование - вычисление погрешности	Апрель 2023
12	создание презентации проекта	Апрель 2023
13	презентация	Апрель 2023

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В мире есть люди, которые не хотят, чтобы кто-то посторонний читал их личные сообщения. По опросу учащихся класса 9м: 80% учащихся (24 человека из 30) хотят, чтобы их сообщения оставались в секрете. Также в данное время военные передают информацию друг другу по рации или через письма, что является не очень надёжным способом.

ЦЕЛЬ И ЗАДАЧИ

Цель: Создать функциональное и удобное приложение-кодировщик для шифрования информации.

Задачи:

1. Изучить существующие методы и алгоритмы шифрования для определения наиболее подходящего подхода к разработке приложения.

2. Создать пользовательский интерфейс приложения, простотой и понятный в использовании.

3. Реализовать функционал шифрования и дешифрования данных в приложении, используя выбранные алгоритмы и методы.

4. Протестировать приложение, проведя тестирование функций шифрования и дешифрования.

РАЗДЕЛ1-ТЕОРИЯ

Tkinter- это графическая библиотека, позволяющая создавать программы с оконным интерфейсом. Эта библиотека является интерфейсом к популярному языку программирования Python

Unicode

Unicode — стандарт кодирования символов, включающий в себя знаки почти всех письменных языков мира. В настоящее время стандарт является преобладающим в Интернете.

ASCII

ASCII - стандарт кодирования знаков латинского алфавита, цифр, некоторых специальных знаков и управляющих последовательностей, принятый в 1963 году Американской ассоциацией стандартов как основной способ представления текстовых данных в ЭВМ.

UTF-8

UTF-8 - распространённый стандарт кодирования символов, позволяющий более компактно хранить и передавать символы Юникода, используя переменное количество байт (от 1 до 4), и обеспечивающий полную обратную совместимость с 7-битной кодировкой ASCII. Стандарт UTF-8 официально закреплён в документах RFC 3629 и ISO/IEC 10646 Annex D.

РАЗДЕЛ2-ПРАКТИЧЕСКАЯ ЧАСТЬ

Мною было разработано приложение “кодировщик Валера” написанное на языке программирования Python (его я выбрал, потому что Python очень практичен и лёгок в освоении. Python подходит для решения задач в которых важна точность результата, а не его оформление). Моё приложение закодирует написанное вами сообщение в зашифрованный вид. Я выбрал систему кодировки Unicode, потому что оно является стандартом в интернете и включающий в себя знаки почти всех письменных языков мира.

Для этого я создал словарь, в котором к каждому символу из русского алфавита присвоил значение цифр из системы кодирования Unicode. В дальнейшем он будет использоваться для кодирования и декодирования сообщений пользователя. Следующим шагом идёт функция encode_text, которая будет кодировать текст. Следующим действием я присвоил пустую строку к переменной и перебрал символы в исходном тексте. После того как программа перебрала все символы она определяет, нужно ли его закодировать. Дальше все символы, не нуждающиеся в кодировке, пишутся так, как есть. После обязательно идёт функция decoded_text, которая будет декодировать текст. Теперь присваиваем пустую строку и создаем переменную, которой присваиваем значение 0. Далее мы проверяем, является ли текущая часть текста закономерностью символов. После этого мы декодируем и добавляем символ в перекодированный текст. Если символ не декодируется, он добавляется как есть. Дальше идёт функция encode_button_click, которая обрабатывает нажатие кнопки ”Кодировать”. Получая введённый текст, мы выполняем функцию encode_text, после отчищаем результат и выводим его. Теперь рассмотрим функцию decode_button_click, которая уже обрабатывает кнопку ”Раскодировать”. После получения закодированного текста декодируем его. Потом отчищаем результат и выводим его. Теперь можно проработать графический интерфейс программы, с помощью библиотеки tkinter. Сначала мы устанавливаем заголовок окна, после чего создаём метки для ввода текста и размещаем элемент в окне. Создаем поле для ввода текста, нуждающегося в кодировке/декодировке. Создаём и размещаем кнопку для кодировки текста и следом добавляем кнопку для декодирования текста. Создаем поле для вывода результата и размещаем его. В заключение запускаем графический цикл интерфейса.

Запускаем графический цикл интерфейса.

КАК ОНО РАБОТАЕТ ОТ ЛИЦА ПОЛЬЗОВАТЕЛЯ

1)В поле сверху впишите предложение для кодировки.
2)Нажмите на кнопку”закодировать” и получите результат в строке снизу.
3)Для перевода текста в привычный вид, впишите декодированный текст в поле сверху.
4)Нажмите на кнопку “раскодировать” и получите результат в строке снизу.

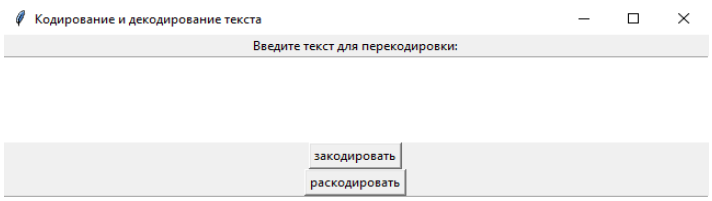

Рис.1-графический интерфейс

Такой текст на входе получит Дуня(рис.2):

```
Перекодированный текст:
043F044004380432043204350442 04340443043D044F
```

Рис.2-закодированный текст

А такой после перевода(рис.3):

```
Раскодированный текст:
привет дуня
```

Рис.3-разкодированный текст

Сравнение кодировки вручную и через приложение

Сначала я вручную закодировал на бумаге одну из цитат Гоголя «Мошенник на мошеннике сидит и мошенником погоняет.», её вы можете видеть на рисунке 4.

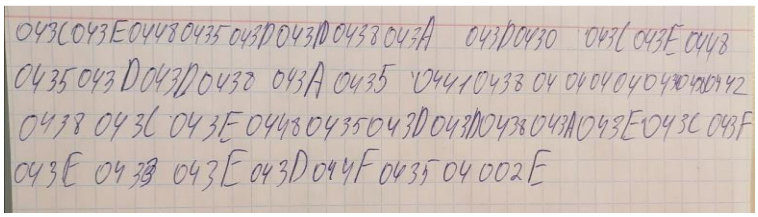

рис.4-закодированный текст

Я закодировал его вручную и сравнил с тем что вывела программа(рис.5).

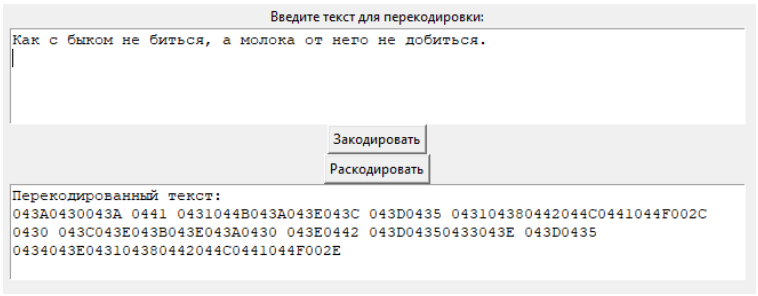

рис.5-закодированный текст в приложении

Дальше я решил раскодировать другое написанное вручную сообщение(рис.6).

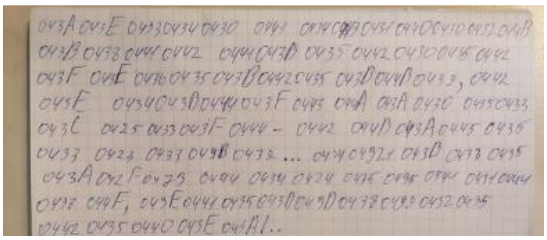

рис.6-закодированный текст

И вот что вывела моя программа (рис.7):

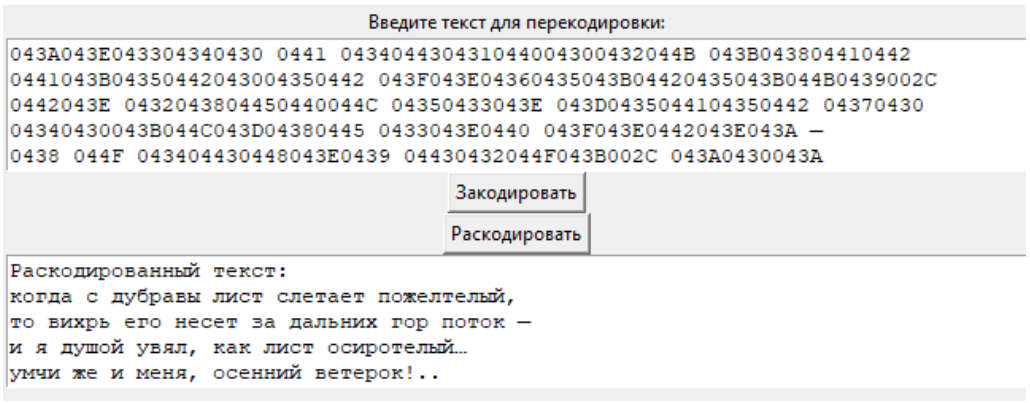

Рис.7-закодированный текст в приложении

Также мне хотелось сравнить, как кодирует моё приложение и сайт из интернета(рис.8).

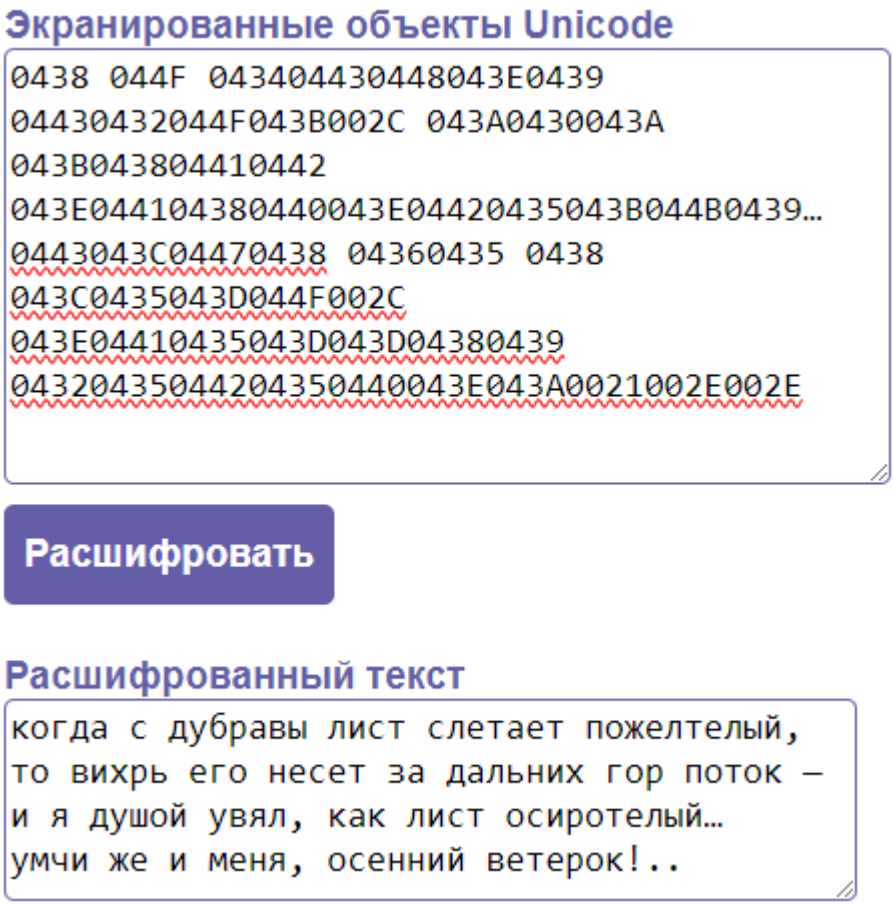

рис.8-закодированный текст на сайте

Сайт успешно расшифровал моё сообщение, и кодировка совпала

Потом мне захотелось проверить, за сколько моя программа закодирует/раскодирует 72881 символов, она справилась за +-1секунду. Сайт потратил на кодировку 3.50секунды и на раскодировку 2 секунды.
Я планирую в дальнейшем создать собственную систему кодировки.

ЗАКЛЮЧЕНИЕ

У меня получилось решить поставленную мной проблему, но также можно ее дорабатывать.

1. придумал оптимальное решение проблемы

2. написал код, который работает без ошибок

3. превратил код в приложение

4. Код в приложение очень гибкий и может легко меняться

5. Приложение уже прошло первые испытания среди моих знакомых, и оно работает офлайн.

СПИСОК ЛИТЕРАТУРЫ

Электронные ресурсы [PytonRu]Электроника;МФТИ. Режим доступа к ресурсу: https://pythonru.com/uroki/obuchenie-python-gui-uroki-po-tkinter Дата обращения 17.02.2024

Электронные ресурсы [Векипедия]Электроника;МФТИ. Режим доступа к ресурсу: https://ru.wikiversity.org/wiki/Курс_по_библиотеке_Tkinter_языка_Python Дата обращения 17.02.2024

Электронные ресурсы [PytonEn]Электроника;МФТИ. Режим доступа к ресурсу: https://docs.python.org/3/library/tkinter.html 17.02.2024

Электронные ресурсы [Xabr]Электроника;МФТИ. Режим доступа к ресурсу:

https://habr.com/ru/articles/133337/ 17.02.2024

Электронные ресурсы [Retyl]Электроника;МФТИ. Режим доступа к ресурсу:

https://www.livelib.ru/author/13161/quotes-nikolaj-gogol 01.04.2024

Код

```
import tkinter as tk
encoding_dict = {
    'а': '0430',
    'А': '0410',
    'б': '0431',
    'Б': '0411',
    'в': '0432',
    'В': '0412',
    'г': '0433',
    'Г': '0413',
    'д': '0434',
    'Д': '0414',
    'е': '0435',
    'Е': '0415',
    'ё': '0451',
    'Ё': '0401',
    'ж': '0436',
```

'Ж': '0416',
'з': '0437',
'З': '0417',
'и': '0438',
'И': '0418',
'й': '0439',
'Й': '0419',
'к': '043A',
'К': '041A',
'л': '043B',
'Л': '041B',
'м': '043C',
'М': '041C',
'н': '043D',
'Н': '041D',
'о': '043E',
'О': '041E',
'п': '043F',
'П': '041F',
'р': '0440',
'Р': '0420',
'с': '0441',
'С': '0421',
'т': '0442',
'Т': '0422',
'у': '0443',
'У': '0423',
'ф': '0444',
'Ф': '0424',
'х': '0445',
'Х': '0425',
'ц': '0446',
'Ц': '0426',
'ч': '0447',
'Ч': '0427',
'ш': '0448',
'Ш': '0428',
'щ': '0449',
'Щ': '0429',
'ъ': '044A',
'Ъ': '042A',
'ы': '044B',
'Ы': '042B',
'ь': '044C',
'Ь': '042C',
'э': '044D',
'Э': '042D',
'ю': '044E',
'Ю': '042E',
'я': '044F',
'Я': '042F',
'0': '0030',

```
    '1': '0031',
    '2': '0032',
    '3': '0033',
    '4': '0034',
    '5': '0035',
    '6': '0036',
    '7': '0037',
    '8': '0038',
    '9': '0039'
}
def encode_text(text):
    encoded_text = ''
    for char in text:
        if char.lower() in encoding_dict:
            if char.isupper():
                encoded_text += encoding_dict[char.lower()].upper()
            else:
                encoded_text += encoding_dict[char]
        else:
            encoded_text += char
    return encoded_text
def decode_text(text):
    decoded_text = ''
    i = 0
    while i < len(text):
        if text[i:i+4] in encoding_dict.values():
            decoded_text                                                    +=
list(encoding_dict.keys())[list(encoding_dict.values()).index(text[i:i+4])]
            i += 4
        else:
            decoded_text += text[i]
            i += 1
    return decoded_text
def encode_button_click():
    input_text = input_text_area.get("1.0", "end-1c")
    encoded_text = encode_text(input_text)
    result_text_area.delete("1.0", "end")
    result_text_area.insert("1.0", "Перекодированный текст:\n" + encoded_text)
def decode_button_click():
    input_encoded_text = input_text_area.get("1.0", "end-1c")
    decoded_text = decode_text(input_encoded_text)
    result_text_area.delete("1.0", "end")
    result_text_area.insert("1.0", "Раскодированный текст:\n" + decoded_text)
root = tk.Tk()
root.title("Кодирование и декодирование текста")
input_label = tk.Label(root, text="Введите текст для перекодировки:")
input_label.pack()
input_text_area = tk.Text(root, height=5, wrap="word")
input_text_area.pack()
```

```
encode_button = tk.Button(root, text="Закодировать", command=encode_button_click)
encode_button.pack()
decode_button = tk.Button(root, text="Раскодировать", command=decode_button_click)
decode_button.pack()
result_text_area = tk.Text(root, height=5, wrap="word")
result_text_area.pack()
root.mainloop()
```