

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Факультет «Информатика и вычислительная техника»

Кафедра «Кибербезопасность информационных систем»

**КУРСОВАЯ РАБОТА**

Тема: «СОЗДАНИЕ ANDROID ПРИЛОЖЕНИЯ С ЗАКОДИРОВАННОЙ БАЗОЙ ДАННЫХ И ИСПОЛЬЗОВАНИЕМ АЛГОРИТМОВ СЖАТИЯ ДАННЫХ»

Дисциплина: «Теория информации»

Специальность: 10.05.01 Компьютерная безопасность

Специализация: Математические методы защит информации

Обозначение курсового работы ТИ.730000.000 Группа ВКБ31

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Т.Г. Закарян

подпись, дата

Курсовая работа защищена с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ст. преподаватель, И.А. Алферова

подпись, дата

Ростов-на-Дону

2022



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Факультет Информатика и вычислительная техника

Кафедра Кибербезопасность информационных систем

**ЗАДАНИЕ**

на выполнение курсовой работы

Тема «СОЗДАНИЕ ANDROID ПРИЛОЖЕНИЯ С ЗАКОДИРОВАННОЙ БАЗОЙ ДАННЫХ И ИСПОЛЬЗОВАНИЕМ АЛГОРИТМОВ СЖАТИЯ ДАННЫХ»

Дисциплина: Теория информации

Обучающийся: Закарян Тигран Геворгович

Обозначение курсовой работы ТИ.730000.000 Группа: ВКБ31

Срок представления работы к защите «\_\_\_» \_\_\_\_\_\_\_\_\_ 20\_\_ г.

Исходные данные для курсовой работы:

1. [С. М. Рацеев](https://www.litres.ru/s-m-raceev/): Элементы высшей алгебры и теории кодирования. Учебное пособие для вузов

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание пояснительной записки** | | | |
| Введение:  Актуальность решения задачи. i | | | |
| Разделы основной части: | | | |
| 1. Изучение алгоритма сжатия данных. 2. Разработка программного средства. Обоснование выбора языка и среды разработки. Архитектура и основные методы, классы. 3. Тестирование программного средства. | | | |
| Заключение:  Выводы по результатам разработки | | | |
| Руководитель работы | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  подпись, дата | И.А. Алферова |
|  |  |  |
| Задание принял к исполнению | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  подпись, дата | Т.Г. Закарян |

Содержание

[Введение 5](#_Toc105409598)

[1. Алгоритмы сжатия данных 6](#_Toc105409599)

[1.1 Общие сведения 6](#_Toc105409600)

[1.2 Алгоритм сжатия LZ78 6](#_Toc105409601)

[2. Разработка программного средства 8](#_Toc105409602)

[2.1 Обоснование выбора языка и среды разработки 8](#_Toc105409603)

[2.2 Архитектура программного средства 8](#_Toc105409604)

[2.3 Основные модули и классы 9](#_Toc105409605)

[2.4 Демонстрация программного средства 10](#_Toc105409606)

[3. Тестирование программного средства 14](#_Toc105409607)

[Заключение 18](#_Toc105409608)

[Перечень используемых информационных ресурсов 19](#_Toc105409609)

[Приложение А 20](#_Toc105409610)

[Приложение Б Листинг кода 24](#_Toc105409611)

# **Введение**

Сжать данные без потерь – извечная проблема всех программистов. Несмотря на доступность информационных носителей достаточного объема и передачи данных на высокой скорости, сжатие в некоторых случаях просто незаменимо. К примеру, при пересылке документации по электронной почте, или для экономии трафика. Если существует необходимость в публикации документов на сайте, сжатие данных будет как нельзя кстати. Когда добавление или замена средств хранения затруднительны, встает вопрос экономии пространства на дисках.

Объект исследования: алгоритмы сжатия данных.

Предмет исследования: реализация программного средства с помощью алгоритмов сжатия данных.

Цель работы заключается в создании программного средства с использованием виртуальной базы закодированных данных.

Задачи исследования:

1. Изучение необходимой теории об алгоритмах кодирования данных.
2. Реализация средства сжатия данных.
3. Реализация программного средства.

# **1. Алгоритмы сжатия данных**

## **1.1 Общие сведения**

Алгоритмы сжатия данных делятся на два метода – искажающие и неискажающие. Искажающие методы сжатия могут искажать исходные данные, после чего полное восстановление невозможно. Неискажающие методы сжатия гарантируют, что декодированные данные будут в точности совпадать с исходными.

## **Алгоритм сжатия LZ78**

Алгоритмы этой группы не используют скользящего окна и в словарь по­мещают не все встречаемые при кодировании строки, а лишь "перспективные" с точки зрения вероятности последующего использования. На каждом шаге в словарь вставляется новая фраза, которая представляет собой кон­катенацию одной из фраз словаря, имеющей самое длинное совпадение со строкой буфера, и символа. Следующим за стро­кой буфера, для которой найдена совпадающая фраза, символ является. В отличие от семейства LZ77 в словаре не может быть одинаковых фраз.

Кодер порождает только последовательность кодов фраз. Каждый код состоит из индекса фразы, префикса, и самого символа. В начале обработки словарь пуст. Далее, теоретически, словарь может расти бесконечно, т. е. на его рост сам алгоритм не налагает ограничений. На практике при достижении определенного объема занимаемой памяти словарь должен очищаться полностью или частично. Также интересным свойством LZ78 является то, что если исходные дан­ные порождены источником с определенными характеристиками, то коэффициент сжатия при­ближается по мере кодирования к минимальному достижимому.

Таким образом LZ78 ориентируется на данные, которые только будут получены. Алгоритм считывает символы сообщения до тех пор, пока накапливаемая подстрока входит целиком в одну из фраз словаря. Как только эта строка перестанет соответствовать хотя бы одной фразе словаря, алгоритм генерирует код, состоящий из индекса строки в словаре, который до последнего введенного символа содержит входную строку, нарушившего совпадение.

# **2.** **Разработка программного средства**

## **2.1 Обоснование выбора языка и среды разработки**

Для данной работы в качестве языка для разработки был выбран Python.

Python – это язык программирования общего назначения, нацеленный в первую очередь на повышение продуктивности самого программиста, нежели кода, который он пишет. Говоря простым человеческим языком, на Python можно написать практически что угодно: веб-приложения, игры, скрипты по автоматизации, комплексные системы расчёта, системы управления жизнеобеспечением и многое другое без ощутимых проблем. Более того, порог вхождения низкий, а код во многом лаконичный и понятный даже тому, кто никогда на нём не писал. За счёт простоты кода, дальнейшее сопровождение программ, написанных на Python, становится легче и приятнее по сравнению с Java или C++. Также Python обладает большим количеством библиотек, технической документации и учебных материалов.

В заключении можно сказать что Python – понятный, простой в изучении, универсальный, востребованный и перспективный язык. Именно поэтому Python является отличным выбором для программиста любого уровня.

## **2.2 Архитектура программного средства**

Архитектура программной системы – это ее организационная структура, включающая модули, их внешние характеристики, а также отношения между модулями.

Требования Платформы (ОС, языки, библиотеки) делится на: производительность, масштабируемость и распределение функциональности по физическим узлам. А также простота поддержки, расширения, повторного использования модулей.

Модульная архитектура программного обеспечения, в которой модули связаны посредством программных интерфейсов, обладает большей гибкостью. Отдельные модули программного компонента можно заменять новыми для адаптации к новым условиям использования, исправления ошибок или добавления новых функций.

При регистрации пользователь вводит свои персональные данные в модальное окно-интерфейс программного средства. После этого пользователь может отправить свои персональные данные путем нажатия кнопки «регистр». С помощью алгоритмов сверточного кодирования и LZ78 данные преобразуются и сохраняются в базе данных.

В случае хранения персональной информации в базе данных, пользователь может авторизироваться в своем личном кабинете. Информация из модальных окон вновь кодируется для того чтобы верифицировать пользователя в системе. Если пользователь найден в системе, то он автоматически переходит в личный кабинет. При неудачной проверке интерфейс остается статичным.

Таким образом получаем систему, использующие технологии и методы разработки, которые будут корректно функционировать в различных мобильных и десктопных операционных системах.

## **2.3 Основные модули и классы**

on\_press: root.manager.current = 'login' – метод переход из одного окна приложения в другое.

class LoginMenu(Screen) – класс с функциями, осуществляющий сверку данных, введенных пользователем, с данными из базы.

con = sqlite3.connect('data1.db') – Модуль sqlite3, метод connect() – подключение к базе данных.

u\_for\_c = self.ids.name\_for\_reg.text – модуль для получения данных из TextInput для осуществления входа в личный кабинет.

self.insert\_varible\_into\_table(username, password) – модуль, который вносит новые закодированные данные в базу.

password = compression(p\_for\_c) – модуль для кодирования данных с помощью алгоритма LZ78.

App().run() – модуль компиляции и запуска приложения.

## **2.4 Демонстрация программного средства**

Программное средство состоит из четырех взаимосвязанных окон: меню, окна регистрации и входа в личный кабинет, личный кабинет. Реализовано с помощью библиотеки kivy на языке python, база данных sqlite3.

Меню состоит из виджетов: двух Button (кнопка), трех Label (текстовое окошко), в которых хранится описание и назначение каждой из кнопок. Первая кнопка переносит пользователя в меню входа в личный кабинет, вторая – в окно регистрации. Результат можно увидеть на рисунке 1.

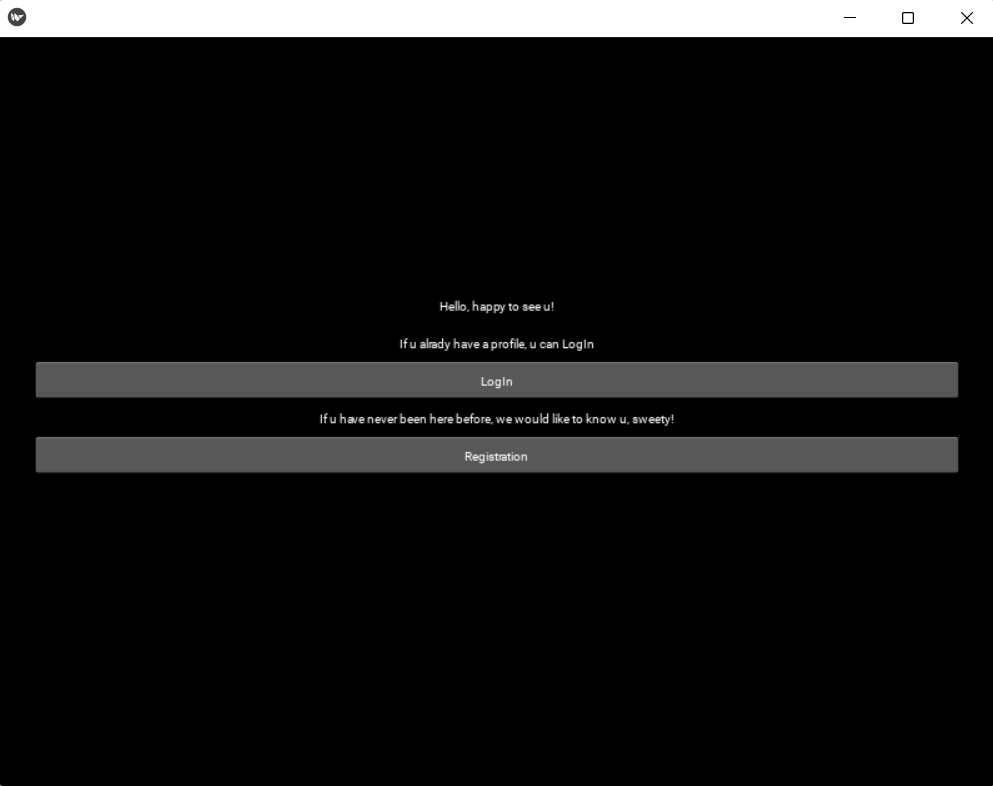


Рисунок 1 – Меню

В окне входа в личный кабинет, как и в окне регистрации, находятся 2 вида виджетов – Button (кнопка) и TextInput (окошко для ввода текста). В окошки для ввода текста пользователь записывает имя и пароль для регистрации или входа, если был зарегистрирован ранее. С помощью первой кнопки программа считывает введенные данные и, при совпадении с базой данных, переносит пользователя в личный кабинет (окно входа) или кодирует и добавляет в базу и переносит юзера назад в меню (окно регистрации). Вторая кнопка возвращает окно меню. Результат можно увидеть на рисунках 2 и 3.

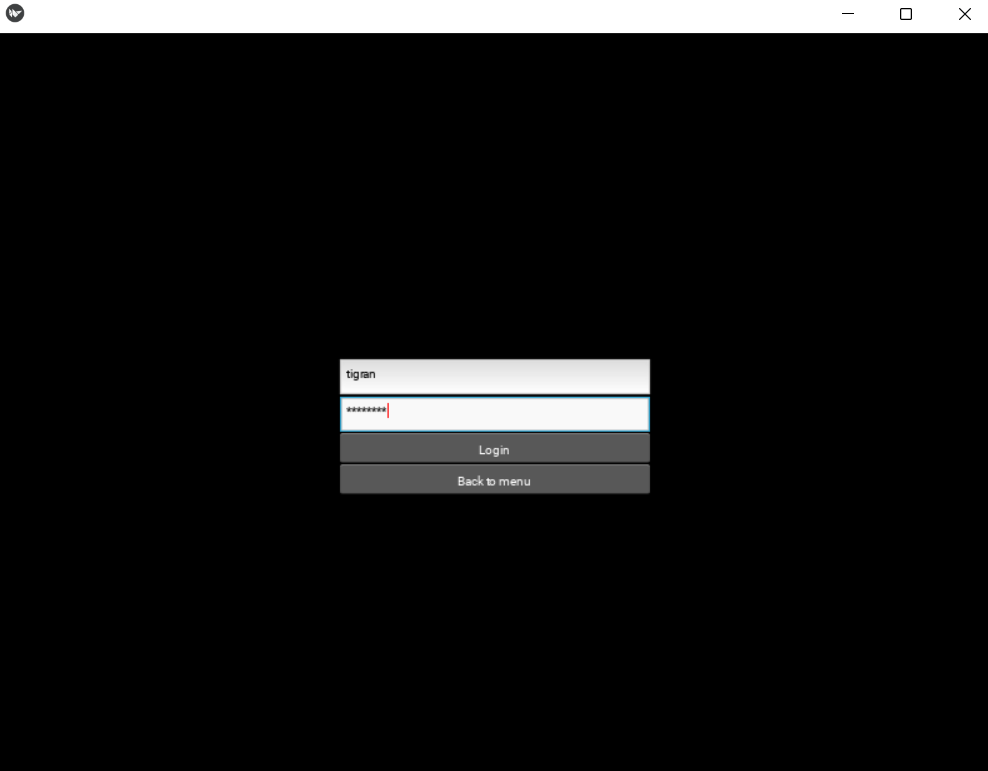


Рисунок 2 – Окно входа в личный кабинет.

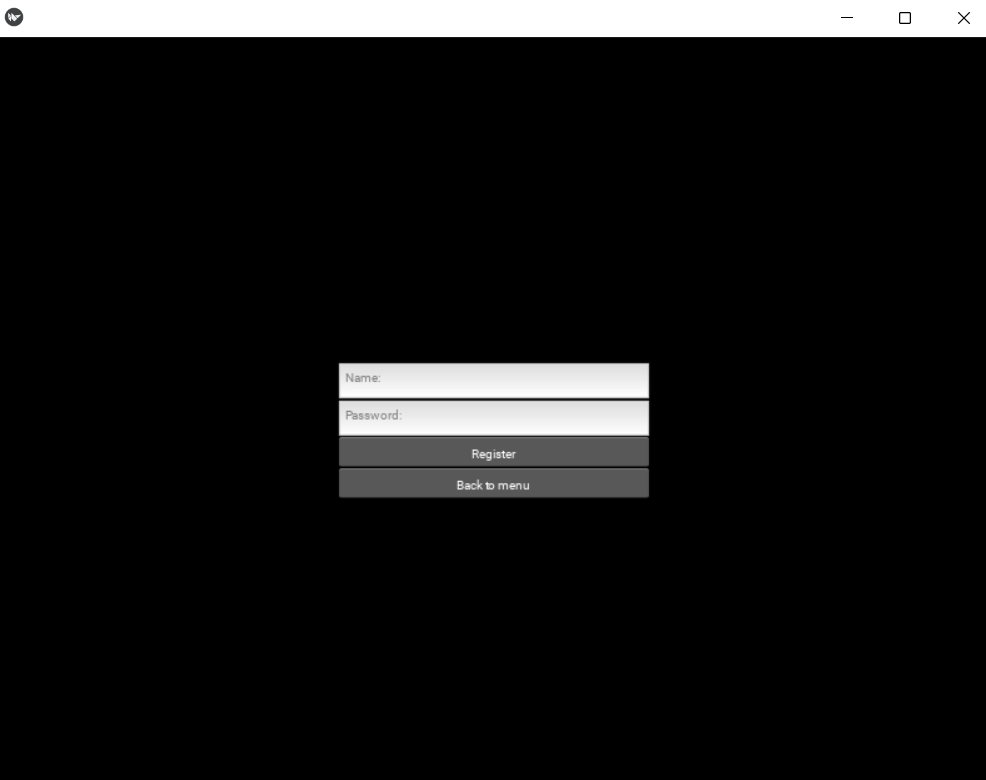


Рисунок 3 – Окно регистрации

Личный кабинет состоит из виджетов: одного Button (кнопка), одного Label (текстовое окошко). Кнопка выходит из личного кабинета и переносит пользователя назад в меню входа в личный кабинет. Результат можно увидеть на рисунке 4.

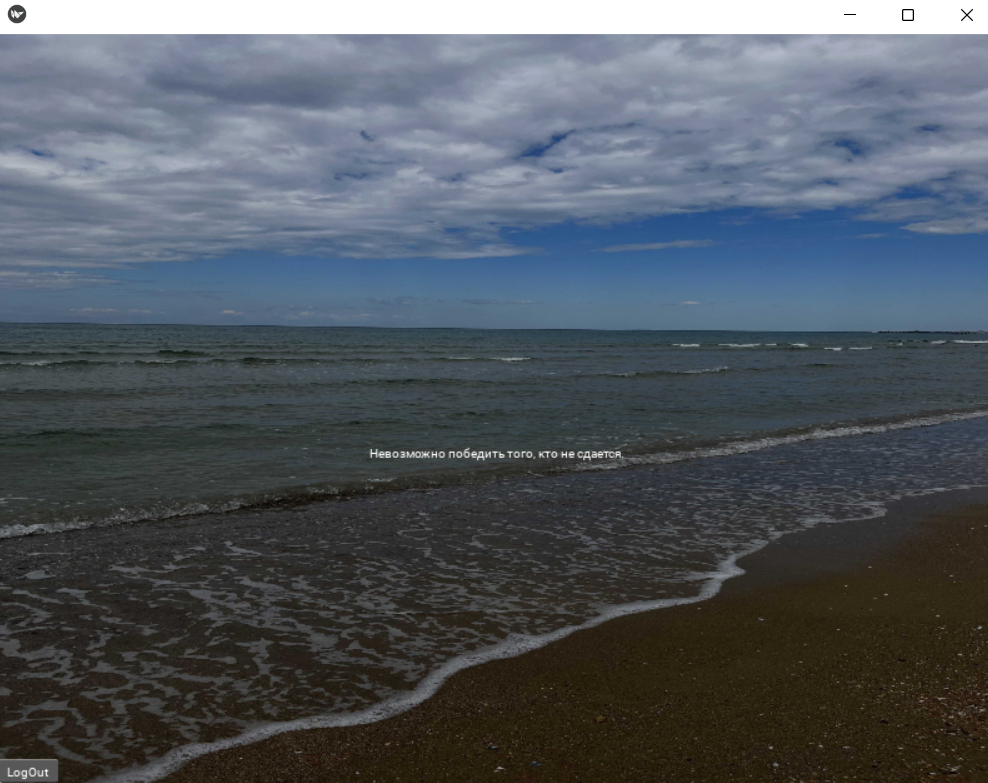


Рисунок 4 – Личный кабинет

Таким образом мы получили графическое представление программного средства, а функции, связанные с работой описаны в других главах.

# **3.** **Тестирование программного средства**

Тестирование заключается в использовании разных данных для проверки работы созданных алгоритмов кодирования.

Тест 1: Текст, представленный на рисунке 5 – случайный набор иностранных букв. Результат выводится в базу данных, что демонстрируется на рисунке 6.

Пароль: abcdefghigklmnopqrstuvwxyz, имя пользователя: abcdefghigklmnopqrstuvwxyz.

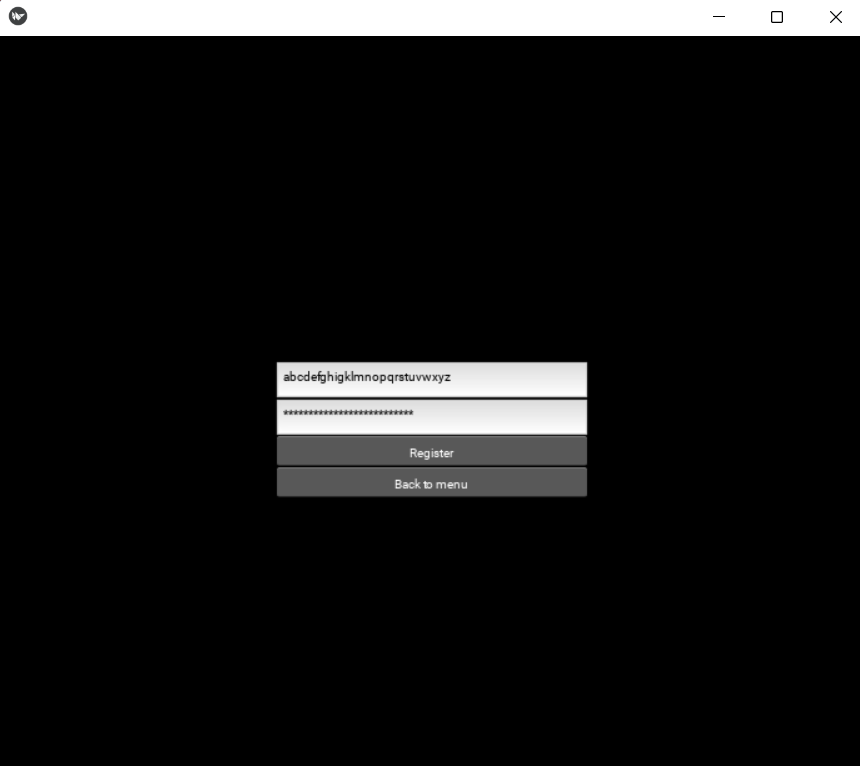


Рисунок 5 – Тест 1

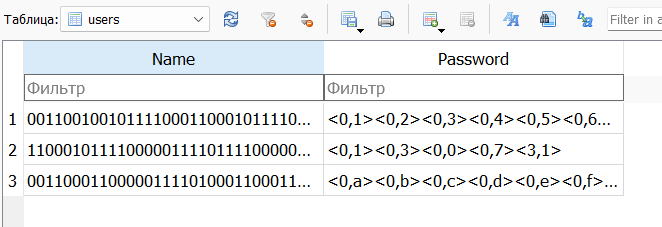


Рисунок 6 – Результат теста 1

Тест 2: Текст, представленный на рисунке 7 – случайный набор букв русского алфавита и цифр. Результат выводится в текстовые окна, как видно на рисунке 8.

Пароль: абвгдеёжзийклмнопрстуфхцчщъыьэюя1234567890, имя пользователя: абвгдеёжзийклмнопрстуфхцчщъыьэюя1234567890.

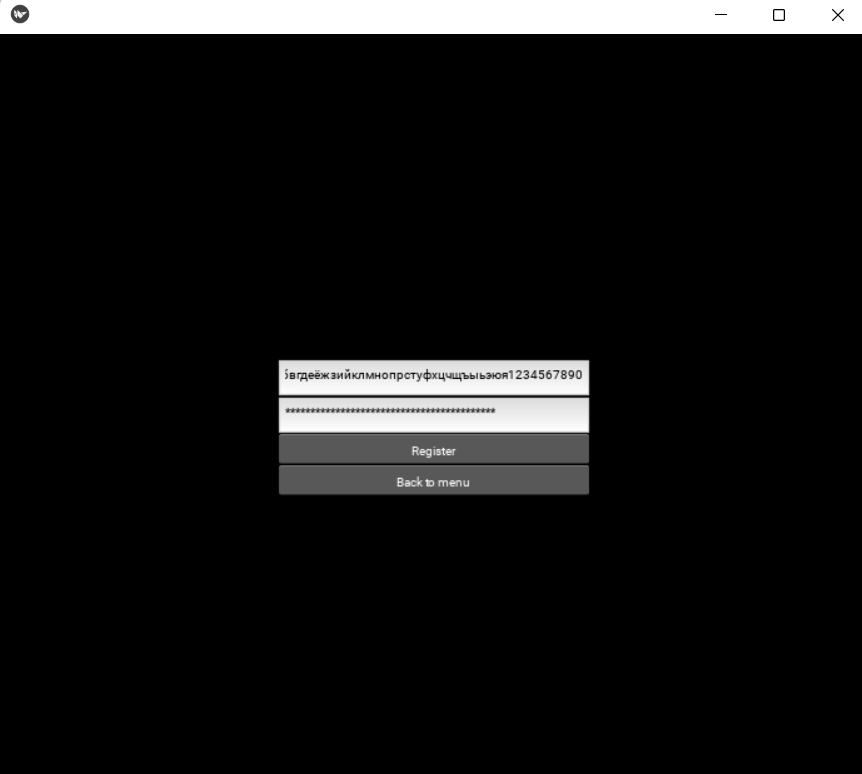


Рисунок 7 – Тест 2

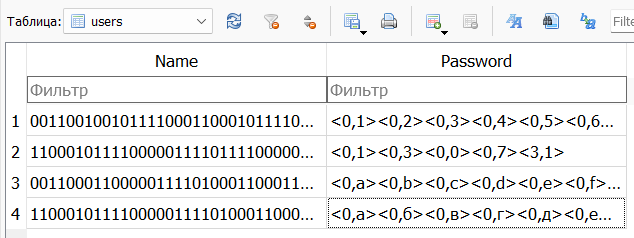


Рисунок 8 – Результат теста 2

Тест 3: Текст, представленный на рисунке 9 – случайный набор букв верхнего регистра русского и английского алфавита, а также символов. На рисунке 10 результат выводится в текстовые окна.

Пароль: ФмНкFqPo!@#$^, имя пользователя: ФмНкFqPo!@#$^.

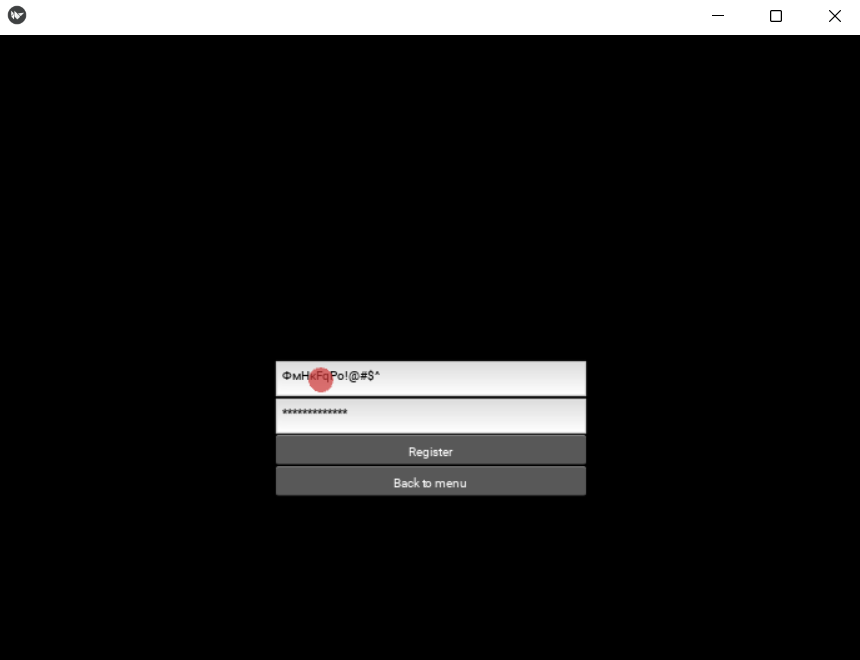


Рисунок 9 – Тест 3

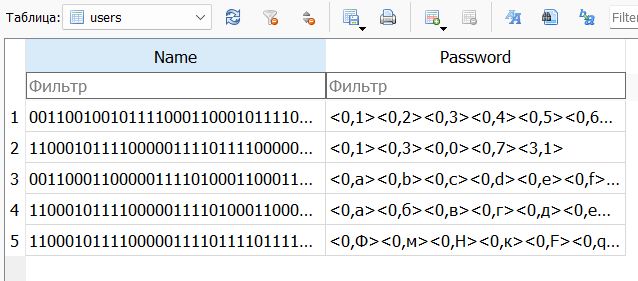


Рисунок 10 – Результат теста 3

Таким образом, можно сделать вывод, что алгоритмы работают корректно, как и все программное средство. Данные всех тестов были успешно внесены в базу и приложение работает с ними корректно.

# **Заключение**

В ходе курсовой работы были рассмотрены такие алгоритмы сжатия данных, как LZ78.

Выполнены следующие поставленные задачи:

1. Изучена необходимая теория об алгоритмах сжатия данных без потерь.
2. Проанализирована работа алгоритма сжатия LZ78.
3. Реализованы алгоритм сжатия LZ78.
4. Реализовано программное средство с базой данных, в которой хранятся результаты работы алгоритмов сжатия данных.

В результате выполнения курсовой работы была достигнута поставленная цель, заключающаяся в создании программного средства, обрабатывающего введенные данные с помощью алгоритма сжатия.

# **Перечень используемых информационных ресурсов**

1. Методы сжатия данных. [Электронный ресурс]. - URL:

<http://www.compression.ru/book/pdf/compression_methods_full_scanned.pdf> (Дата обращения: 07.03.2022)

1. Котенко В.В., Румянцев В.Е., Теория Информации // Издательство Южного федерального университета, 2018г.–241 c.
2. Сверточный кодирование. [Электронный ресурс]. - URL: https://kivy.org/doc/
3. Кудряшов, Б.Д. Теория информации: Учебник для вузов. // СПб.: Питер, 2009г. –320 с.
4. Алгоритмы сжатия данных без потерь [Электронный ресурс] URL: https://habr.com/ru (Дата обращения: 10.04.2022)
5. Учебник Сэломон. Д. Сжатие данных, изображения и звука. // Техносфера, 2006г – 368 с.
6. Ватодин Д., Ратушнях А., Смирнов М., Юкин В. Методы сжатия данных. // ДИАЛОГ-МИФИ, 2002г. - 384 c.

# **Приложение А**

Техническое задание

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| СОГЛАСОВАНО | |  | |  | | «УТВЕРЖДЕНО» |
| Руководитель работы | |  | |  | | И.о. зав каф. «КБИС» |
| \_\_\_\_\_\_\_\_\_ И.А. Алферова |  | |  | |  | \_\_\_\_\_\_\_\_\_ к.т.н. Д.А. Короченцев |
| (подпись) |  | |  | |  | (подпись) |
| «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2022 г. | |  | |  | | «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2022 г. |

**А.1 Введение**

**А.1.1 Наименование программы**

Наименование программы – «Приложение с кодированием персональных данных».

**А.1.2 Область применения**

Областью применения является любое приложение, в котором присутствует раздел для авторизации пользователя.

**А.2 Основания для разработки**

Разработка ведется в рамках выполнения курсовой работы по дисциплине «Теория информации». Тема разработки «Создание Android приложения с закодированной базой данных и использованием алгоритмов сжатия данных».

**А.3 Назначение разработки**

**А.3.1 Функциональное назначение**

Функциональным назначением программного средства является реализация приложения, содержащего алгоритмы кодирования обмена данных

в безопасном формате.

**А.3.2 Эксплуатационное назначение**

Эксплуатационное назначение: программное средство предназначено для эксплуатации в организациях, которым необходимо быстрая и качественная передача данных.

**А.4 Требования к программе**

**А.4.1 Требования к функциональным характеристикам**

Программное средство должно обрабатывать данные, вводимые пользователем с помощью двух алгоритмов кодирования и переносить их в базу данных.

1. convolotional – функция, реализующая алгоритм сжатия данных
2. insert\_varible\_into\_table – функция, реализующая перенос закодированных данных в базу данных

**А.4.2 Требования к надёжности**

Надёжность функционирования обеспечивается корректным функциони-рованием аппаратного обеспечения и правильным выбором входных данных.

**А.4.3 Условия эксплуатации**

Условия эксплуатации совпадают с условиями эксплуатации персональ-ных ЭВМ IBM PC и совместимых с ним персональных компьютеров. Требуется наличие одного оператора. Программа рассчитана на пользователя, владеющего базовыми навыками работы с персональным компьютером. Входные данные – текстовые сообщения. Результат работы не требует пояснений.

**А.4.4 Требования к составу и параметрам технических средств**

Для работы программы требуется наличие IBM PC – совместимого персонального компьютера с тактовой частотой процессора не менее 1,5 Гц.

**А.4.5 Требования к информационной и программной совместимости**

Базовый язык программирования Python 3.7.

**А.4.6 Требования к маркировке и упаковке**

Требований к маркировке и упаковке не предъявлялось.

**А.4.7 Требования к транспортированию и хранению**

Требования к транспортировке не предъявлялось.

**А.5 Требования к программной документации**

Программная документация должна включать документ «Техническое задание» (ГОСТ 19.201-78).

**А.6 Стадии и этапы разработки**

1. Системный анализ.

2. Общесистемное проектирование.

3. Подготовка технологических средств.

4. Программная реализация, рабочий проект.

5. Отладка программного средства в статике.

6. Тестовые испытания программного комплекса.

**А.7 Порядок контроля и приемки**

Контроль и приемка разработки осуществляется на основе испытаний контрольно-отладочных примеров. Примеры должны демонстрировать правильность работы используемых в программе структур данных и алгоритмов в различных ситуациях, которые могут возникнуть при выполнении программы. При этом проверяются выполнение всех функций программы и полнота документации.

|  |  |
| --- | --- |
| **Разработчик:** | Закарян Тигран Геворгович |
| **Дата начала разработки:** | «10» февраля 2022 г. |

# **Приложение Б Листинг кода**

project.py:

Builder.load\_string("""

<MenuScreen>:

BoxLayout:

orientation: 'vertical'

padding: [30,200,30,250]

Label:

text: 'Hello, happy to see u!'

font\_size: 10

size: 250, 20

Label:

text: 'If u alrady have a profile, u can LogIn'

font\_size: 10

size: 250, 20

Button:

text: 'LogIn'

font\_size: 10

size: 250, 20

on\_press: root.manager.current = 'login'

Label:

font\_size: 10

size: 250, 20

text: 'If u have never been here before, we would like to know u, sweety!'

Button:

text: 'Registration'

font\_size: 10

size: 250, 20

on\_press: root.manager.current = 'registration'

<RegistrationMenu>:

BoxLayout:

orientation: 'vertical'

padding: [30,230,30,230]

TextInput:

id: name\_for\_reg

font\_size: 10

size: 250, 30

size\_hint: None,None

pos\_hint: {'center\_x': 0.5}

hint\_text: 'Name:'

multiline: False

TextInput:

id: pass\_for\_reg

font\_size: 10

size: 250, 30

size\_hint: None,None

pos\_hint: {'center\_x': 0.5}

hint\_text: 'Password:'

password: True

multiline: False

Button:

text: 'Register'

font\_size: 10

size: 250, 25

size\_hint: None,None

pos\_hint: {'center\_x': 0.5}

on\_press: root.registration()

Button:

text: 'Back to menu'

font\_size: 10

size: 250, 25

size\_hint: None,None

pos\_hint: {'center\_x': 0.5}

on\_press: root.manager.current = 'menu'

<LoginMenu>:

BoxLayout:

orientation: 'vertical'

padding: [30,230,30,230]

TextInput:

id: login

font\_size: 10

size: 250, 30

size\_hint: None,None

pos\_hint: {'center\_x': 0.5}

hint\_text: 'Name:'

multiline: False

TextInput:

id: password

font\_size: 10

size: 250, 30

size\_hint: None,None

pos\_hint: {'center\_x': 0.5}

hint\_text: 'Password:'

password: True

multiline: False

Button:

text: 'Login'

font\_size: 10

size: 250, 25

size\_hint: None,None

pos\_hint: {'center\_x': 0.5}

on\_press: root.login()

Button:

text: 'Back to menu'

font\_size: 10

size: 250, 25

size\_hint: None,None

pos\_hint: {'center\_x': 0.5}

on\_press: root.manager.current = 'menu'

<PS>:

FloatLayout:

id:fl1

orientation: 'vertical'

canvas:

Rectangle:

source: 'fon2.jpg'

size: self.size

pos: self.pos

BoxLayout:

orientation: 'vertical'

padding: [250,250]

Label:

font\_size: 10

size: 250, 30

text: 'Невозможно победить того, кто не сдается.'

size\_hint: None,None

pos\_hint: {'center\_x': 0.5}

BoxLayout:

orientation: 'vertical'

Button:

text: 'LogOut'

font\_size: 10

size: 50, 20

size\_hint: None,None

pos\_hint: {'left\_x': 0.5}

on\_press: root.manager.current = 'menu'

""")

turbo.py:

def compression(string):

dict = {}

entry = ''

index = 1

for i in range(len(string)):

entry += string[i]

if entry not in dict:

lst = [index]

encoder = [dict[entry[0:len(entry) - 1]][0] if entry[0:len(entry) - 1] in dict else 0, entry[-1]]

lst.append(encoder)

dict[entry] = lst

index += 1

entry = ''

ans = ''

for x in dict:

ans += f'<{dict[x][1][0]},{dict[x][1][1]}>'

return ans

def parse(string):

dict = {}

index = 1

incorrect = False

for i in range(len(string)):

if string[i] == '<':

encoderIndex = ''

encoderTail = ''

comma = False

i += 1

while string[i] != '>':

if string[i] == ',':

comma = True

i += 1

continue

if comma:

encoderTail += string[i]

i += 1

continue

encoderIndex += string[i]

i += 1

lst = [[int(encoderIndex), encoderTail], '']

dict[index] = lst

index += 1

return dict

def decompression(string):

error = False

ans, entry = '', ''

try:

parse(string)

except BaseException:

error = True

return error, ans, {}

dict = parse(string)

for x in dict:

value = dict[x]

entry += dict[value[0][0]][1] if value[0][0] != 0 else ''

entry += value[0][1]

value[1] = entry

ans += entry

entry = ''

return ans