**С****АНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики-процессов управления**

**Программа бакалавриата**

**“Большие данные и распределенная цифровая платформа”**

**ОТЧЕТ**

**по лабораторной работе №2**

**по дисциплине «Алгоритмы и структуры данных»**

**на тему «Обезличивание данных»**

**Вариант – 1**

**Студент гр. 23Б15-пу**

**Черевко М.Е.**

**Преподаватель**

**Дик А.Г.**

**Санкт-Петербург**

**2024 г.**

Оглавление

1. [Цель работы 3](#_Toc180702635)
2. [Описание задачи (формализация задачи) 4](#_Toc180702636)
3. [Теоретическая часть 5](#_Toc180702637)
4. [Ход программы 6](#_Toc180702638)
5. [Блок схема программы 8](#_Toc180702639)
6. [Описание программы 11](#_Toc180702640)
7. [Рекомендации пользователя 15](#_Toc180702641)
8. [Рекомендации программиста 17](#_Toc180702642)

[**Код программы:** 17](#_Toc180702643)

1. [Контрольный пример 18](#_Toc180702644)
2. [Вывод 21](#_Toc180702645)
3. [Источники 22](#_Toc180702646)

# Цель работы

Цель данной работы заключается в разработке программы для **обезличивания** большого объёма данных (датасета), имитирующего сценарий покупки в магазине. Обезличенный датасет должен сохранить полезную информацию, включать различные атрибуты, такие как название магазина, его местоположение, дату и время, категории товаров этого магазина и соответствующие бренды, номер платёжной карты, через которую была совершена оплата, наименования банка и платёжной системы, число покупок и общая стоимость.

# Описание задачи (формализация задачи)

1. Программа должна считывать файл, который является таблицей exel или таблицей другого формата. Входной файл представляет собой датасет с информацией покупок в магазинах. Основная задача программы — обезличивание данных этого датасета. Для успешной работы программы полученная таблица должна иметь следующие столбцы: **Магазин, Широта, Дата и время, Категория, Бренд, Номер карты, Банк, Платежная система, Количество товаров, Стоимость**
2. Программа должна быть разделена на модули, где каждый отвечает за отдельный функционал:

* Обезличивание входного набора данных с использованием различных методов (агрегация, маскеризация, и т.д.).
* Вычисление степени К-анонимности для входного набора данных.

1. Пользователь может указывать квази-идентификаторы, на основе которых будет проводиться обезличивание и оценка К-анонимности.
2. После обезличивания программа должна рассчитать показатель К-анонимности для каждого из наборов данных и вывести "плохие" значения (с низким К).
3. Программа должна анализировать уникальные строки в наборе данных и выводить их в случае, если К = 1.

# 

# Теоретическая часть

1. В проекте задействованы следующие методы обезличивания:

* **Агрегация** — это метод обезличивания данных, при котором несколько исходных значений группируются в одно обобщённое значение для уменьшения детальности данных.
* **Микро-агрегация** — это метод обезличивания, при котором значения атрибутов разделяются на небольшие группы, и для каждой группы вычисляется обобщённое значение, которое затем присваивается всем элементам группы.
* **Локальное обобщение** — это метод обезличивания, при котором конкретные значения заменяются на более обобщённые категории, при этом степень обобщения варьируется в зависимости от контекста.
* **Маскеризация** — это метод обезличивания, при котором исходные значения заменяются специальными символами или псевдозначениями, которые скрывают оригинальную информацию, например в представленной программе код банковской карты маскируется символом \*

1. Так же разработан расчет **К-анонимности** для отображение степени защищённости. **K-анонимность** — это метод защиты данных, обеспечивающий, что каждая запись в наборе данных не может быть однозначно идентифицирована менее чем среди **k** других записей. Это достигается за счёт группирования записей с одинаковыми квази-идентификаторами, такими как возраст, пол, место жительства, что минимизирует риск раскрытия личности конкретного индивида в обезличенных данных.
2. **Полезность данных**: обезличивание происходит таким образом, что данные сохраняют ценную информацию. Это позволяет проводить анализ тенденций и закономерностей, не раскрывая при этом личные данные. Например, отфильтрованная информация по категориям и дате могут помочь в выявлении популярных товаров или сезонных колебаний спроса.

# Ход программы

1. **Запуск программы и импорт необходимых библиотек**:

* Импорт библиотек: pandas, tkinter, os.
* Объявление и определение функций.
* Программа начинается с создания графического интерфейса с помощью Tkinter, где пользователь может загружать файлы, обезличивать данные и проверять K-анонимность.
* Основные элементы интерфейса: кнопки для загрузки файла, анонимизации данных, проверки K-анонимности и выхода.

1. **Загрузка файла** (load\_file()):

* При нажатии кнопки "Загрузить файл" вызывается функция load\_file(), которая позволяет пользователю выбрать файл Excel или CSV.
* Файл загружается в DataFrame df с использованием pandas. В случае ошибки пользователю показывается соответствующее сообщение.

1. **Выбор квази-идентификаторов** (choose\_quasi\_identifiers()):

* Пользователь может выбрать квази-идентификаторы (столбцы данных) для анонимизации.
* Открывается окно с чекбоксами, где можно выбрать необходимые квази-идентификаторы. После подтверждения выбор передается в функцию anonymize\_data().

1. **Процесс анонимизации данных** (anonymize\_data(selected\_identifiers)):

* В зависимости от выбранных квази-идентификаторов, вызываются следующие функции:
  + replace\_coordinates\_with\_city(df) — заменяет географические координаты на города.
  + aggregate\_date\_season(df) — агрегирует даты по годам и сезонам, подсчитывая количество транзакций в каждом сезоне.
  + suppress\_card\_numbers(df) — маскирует номера карт (заменяет их на \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*).
  + aggregate\_price(df) — микро-агрегирует цену товаров, заменяя стоимость минимальной и максимальной ценой за единицу.
  + aggregate\_items(df) — суммирует количество товаров в рамках каждой группы (Магазин, Категория, Бренд).
  + aggregate\_banks(df) — агрегирует данные банков, подсчитывая количество транзакций для каждого банка.
  + aggregate\_payment\_systems(df) — агрегирует платёжные системы, подсчитывая количество транзакций для каждой системы.
* После успешного завершения анонимизации данных вызывается функция save\_anonymized\_data() для сохранения изменённого файла.

1. **Проверка K-Анонимности** (check\_k\_anonymity()):

* По нажатию кнопки "Проверить K-Анонимность" программа запускает функцию проверки.
* calculate\_k\_anonymity(df, quasi\_identifiers\_k) — вычисляет K-анонимность на основе всех столбцов таблицы. Возвращает DataFrame с количеством уникальных записей (различных комбинаций значений) для каждого набора квази-идентификаторов и минимальное значение K.
* Если минимальное значение K равно 1, то программа сообщает об уникальных строках, что указывает на низкий уровень анонимности.
* Результаты проверки выводятся пользователю.

1. **Завершение программы** (quit\_program()):

* Нажатие кнопки "Выйти" вызывает подтверждающий запрос на завершение программы и закрывает окно Tkinter при положительном ответе.
* Завершение программы извне

# Блок схема программы

Рис 1. Блок-схема основной программы

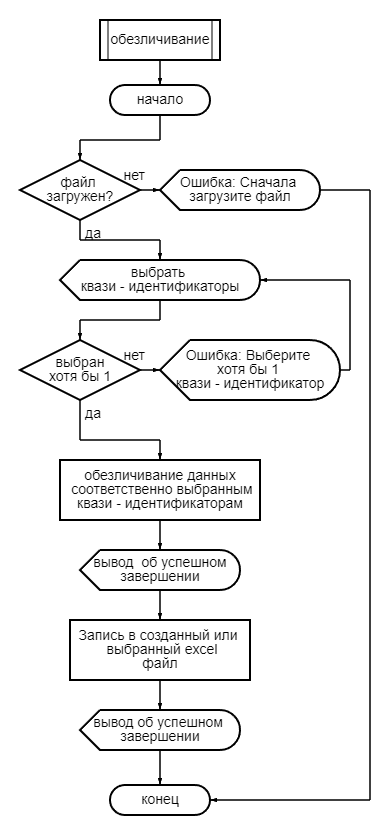


Рис 2. Блок – схема **обезличивания** датасета

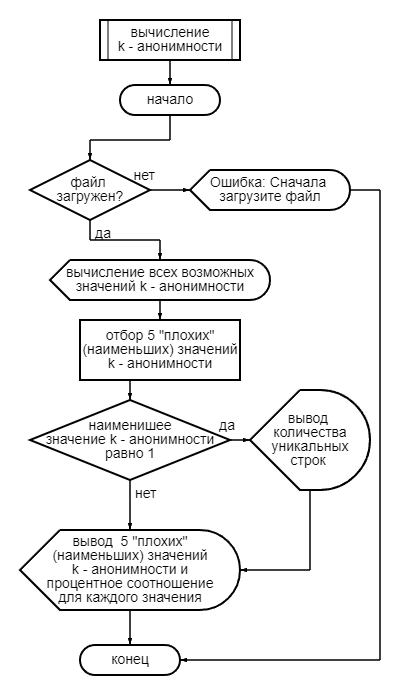


Рис. 3 Блок – схема вычисления k - анонимности

# 

# Описание программы

В данном проекте реализована программа на языке Python (версия 3.11.9), которая принимает набор данных, хранящий сведенья о покупках в различных магазинах в виде таблицы (состоящей из столбцов: **Магазин, Широта, Долгота, Дата и время, Категория, Бренд, Номер карты, Банк, Платежная система, Количество товаров, Стоимость**) и **обезличивает**. Программа использует библиотеки, **pandas**, **tkinter** и **os** для работы с данными, создания интерфейса и Excel-файлов.

Таблица depersonalization\_functions

|  |  |  |
| --- | --- | --- |
| Функция | Описание | Возвращаемое значение |
| load\_file() | Загружает файл Excel или CSV, используя диалоговое окно. | DataFrame с данными файла |
| replace\_coordinates\_with\_city(df) | Заменяет географические координаты на города. | DataFrame с заменёнными координатами |
| aggregate\_date\_season(df) | |  | | --- | |  |  |  | | --- | | Агрегирует даты по годам и сезонам, подсчитывая количество транзакций за каждый сезон. | | DataFrame с агрегированными датами |
| suppress\_card\_numbers(df) | Маскирует номера карт, заменяя их на \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*. | DataFrame с маскированными номерами карт |
| aggregate\_items(df) | Суммирует количество товаров по группам (Магазин, Категория, Бренд). | DataFrame с агрегированными товарами |
| aggregate\_price(df) | Выполняет микро-агрегацию цен товаров, указывая минимальную и максимальную стоимость за единицу. | DataFrame с агрегированными ценами |
| aggregate\_banks(df) | Подсчитывает количество транзакций для каждого банка. | DataFrame с агрегированными банками |
| aggregate\_payment\_systems(df) | Подсчитывает количество транзакций для каждой платёжной системы. | DataFrame с агрегированными системами |
| save\_depersonalized\_data() | Сохраняет деперсонализированные данные в файл | Нет возвращаемого значения |
| depersonalize\_data(selected\_identifiers) | Выполняет анонимизацию данных на основе выбранных квази-идентификаторов. | DataFrame с анонимизированными данными |
| choose\_quasi\_identifiers() | Открывает интерфейс для выбора квази-идентификаторов с помощью чекбоксов. | Список выбранных квази-идентификаторов |
| calculate\_k\_anonymity(df, quasi\_identifiers\_k) | Вычисляет K-анонимность для выбранных квази-идентификаторов. | DataFrame с количеством уникальных строк |
| find\_bad\_k\_values(k\_anonymity\_df) | Находит строки с низким уровнем K-анонимности (ниже заданного порога). | DataFrame с плохими значениями K |
| check\_k\_anonymity() | Запускает процесс проверки K-анонимности. | DataFrame с результатами проверки K |
| quit\_program() | Закрывает программу и завершает её выполнение. | Нет возвращаемого значения |

**Структура программы**:

* 1. **Модуль** Операции с файлами

**Функции:**

* load\_file(): Загружает файл из Excel или CSV.
* save\_depersonalized \_data(): Сохраняет обезличенные данные в новый файл.
  1. **Модуль** Обработки данных

**Функции:**

* replace\_coordinates\_with\_city(df): Заменяет координаты городами.
* aggregate\_date\_season(df): Агрегирует даты по годам и сезонам.
* suppress\_card\_numbers(df): Маскирует номера карт.
* aggregate\_items(df): Агрегирует товары по группам.
* aggregate\_price(df): Выполняет микро-агрегацию стоимости товаров.
* aggregate\_banks(df): Агрегирует данные по банкам.
* aggregate\_payment\_systems(df): Агрегирует данные по платёжным системам.
  1. **Модуль** Обезличивания данных

**Функции:**

* depersonalize\_data(selected\_identifiers): Обезличивание данных по выбранным квази-идентификаторам.
* calculate\_k\_anonymity(df, quasi\_identifiers\_k): Вычисление K-анонимности.
* find\_bad\_k\_values(k\_anonymity\_df): Поиск строк с низким значением K-анонимности.
  1. **Модуль** main.py (Управление программой через пользовательский интерфейс)
* **Функция**: choose\_quasi\_identifiers(): Окно для выбора квази-идентификаторов.
* Интерфейс на Tkinter для загрузки и обработки данных.
* Инициализация и запуск приложения.
* Объединение всех процессов: загрузка, анонимизация, агрегация данных, проверка K-анонимности.
* Управление логикой работы программы и вызов интерфейса.

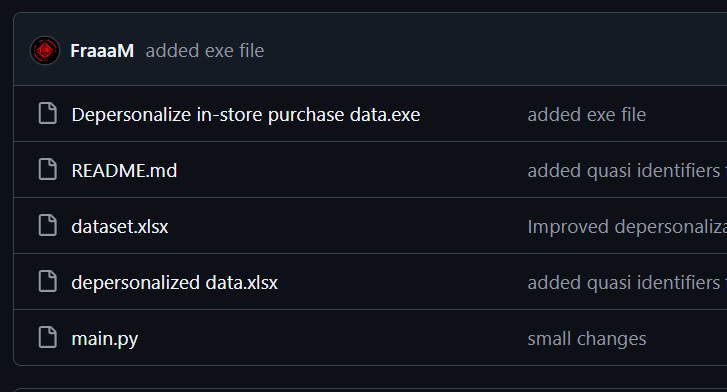
# Рекомендации пользователя

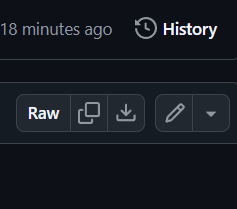
**Инструкция к запуску программы:**

Для успешного использования данной программы пользователю рекомендуется следовать следующим инструкциям:

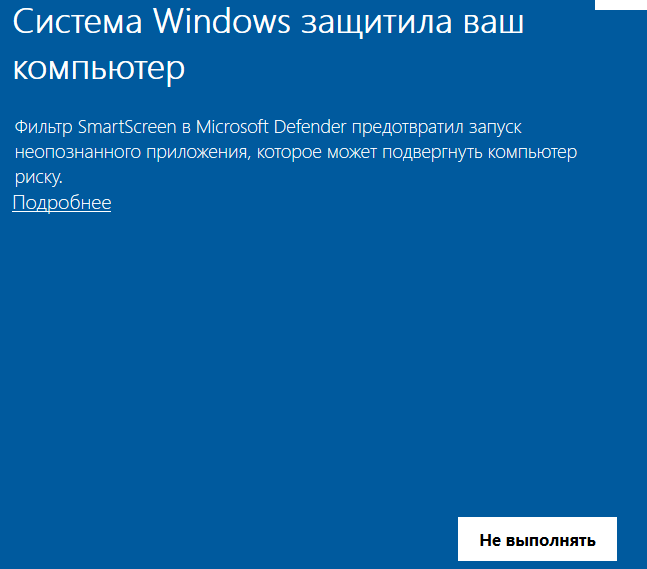
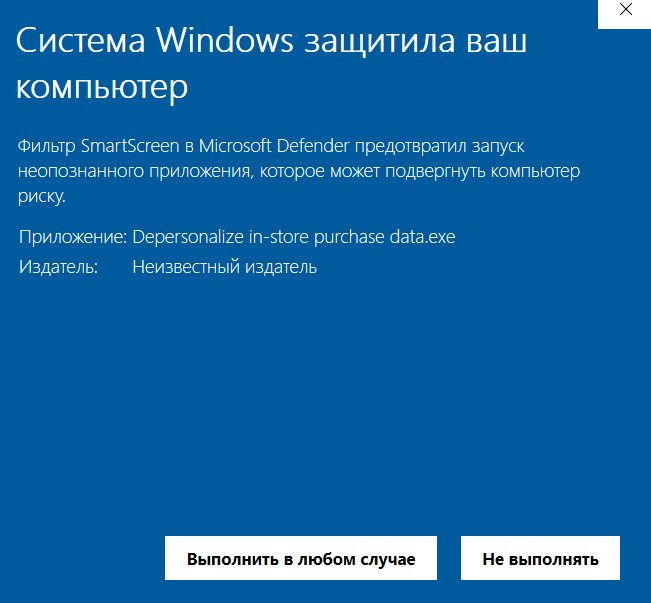
**1. Перейти по ссылке:** [**https://github.com/FraaaM/Depersonalization\_and\_anonymization\_of\_the\_data\_set**](https://github.com/FraaaM/Depersonalization_and_anonymization_of_the_data_set)

**2. Нажать на Depersonalize in-store purchase data.exe, перейдя на другую страницу**



**3. В правом верхнем углу нажать на окно установки файла**

**4. После установки запустите приложение. Если система windows запретила запуск, нажмите Подробнее и Выполнить в любом случае.**

****

* 1. **Пример использования программы и инструкция представлены на страницах 18 – 20.**
  2. **ВАЖНО: Для эффективного обезличивания ваша таблица должна иметь вид (включать такие же названия строк) как таблица dataset.xlsx по ссылке в пункте 1.**
  3. **Изучите для работы с обработанными данных. Обезличивание происходит следующим образом:**

Назовём уникальной группой множество строк (покупок), где столбцы Магазин, Категория, Бренд совпадают у каждого элемента группы (то есть это записи о покупках конкретного товара в конкретном магазине)

Для столбца **Количество товаров** находится общее число покупок товара в группе, и это значение присваивается каждому элементу группы.

Для столбца **Стоимость** находится диапазон цены для каждой группы. Для столбцов **Банк** и **Платёжная система** (п.с.) считается сколько раз был использован каждый банк\п.с. в транзакциях группы (сколько раз встречался у элементов группы). Затем каждому элементу группы записывается банк\п.с. и рядом, в скобках, сколько раз этот банк\п.с. встречался (сколько раз было оплачено через него\п.с.). **Например:** *Сбербанк(8), Тинькофф(4), ВТБ(2).* Значит через сбербанк было оплачено 8 раз и так по аналогии для каждого банка (если какого-то банка нет в списке значит через него не проводилась оплата), тогда всего транзакций: 8 + 4 + 2 = 14.

Столбец **Дата и время** переводится в год и времена года (лето и т.п.) и обезличивается по аналогии как с банком и п.с. **Например**: *2024, Лето(6), Осень(6), Зима(5), Весна(1).* Эта строка означает, что **летом** было **6** покупок конкретного товара в конкретном магазине и так для каждого времени года, тогда за год всего 6+6+5+1 = 18 покупок.

Столбцы **Широта** и **Долгота** заменяются на столбец **Местоположение**, где указаны наименования городов, к которому относятся координаты.

**Код платёжной карты** заменяется на \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Рекомендации программиста

* **Программу можно запустить на любом IDE, поддерживающем Python.**
* **Актуальность библиотек и языка программирования:**

Регулярно обновляйте используемые библиотеки и версии Python, чтобы обеспечить совместимость программы с современными операционными системами и избежать проблем с работоспособностью.

* **Организация кода:**

Структурируйте код, разделяя его на логически обоснованные модули и функции. Это упростит дальнейшую разработку и поддержку программы.

* **Документирование:**

Создавайте и поддерживайте документацию по проекту, чтобы другие разработчики могли быстро понять его структуру и функциональность. Это облегчает совместную работу и сопровождение кода.

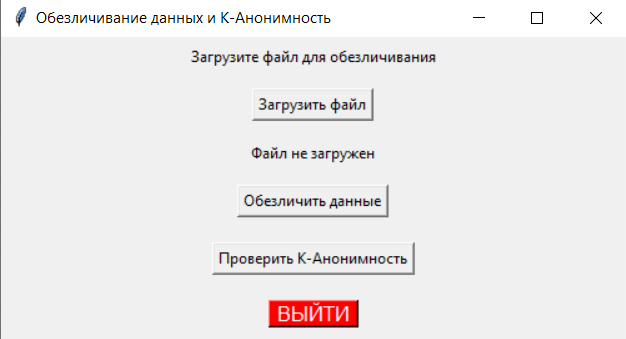
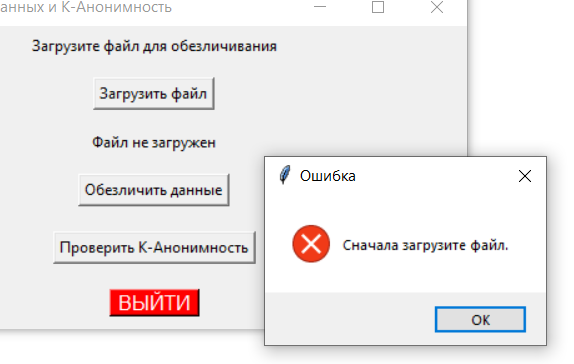
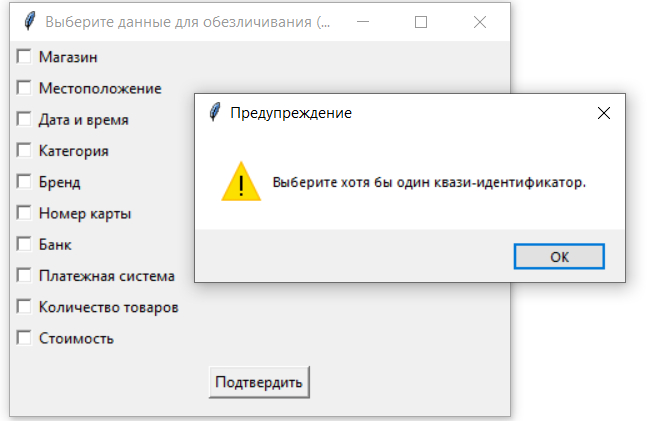
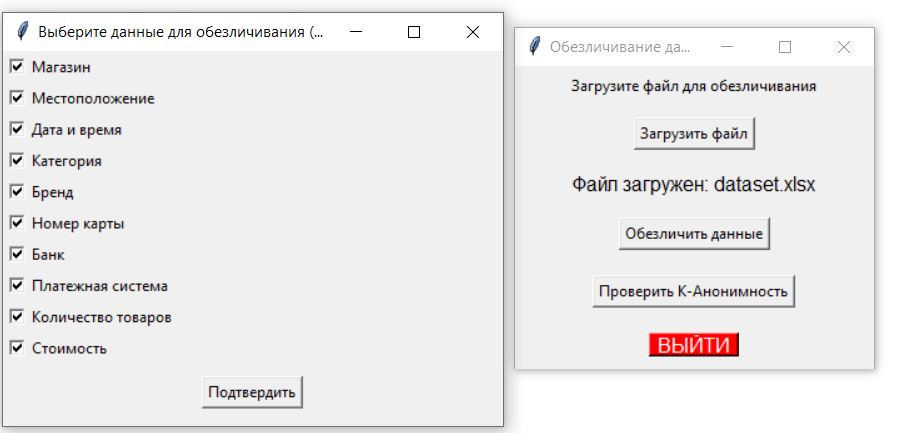
**Код программы:**

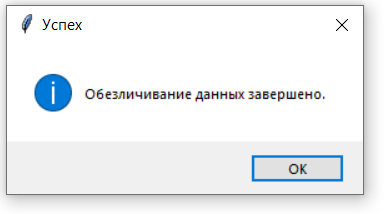
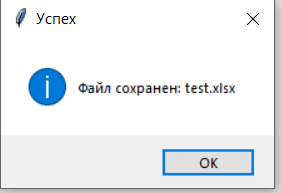
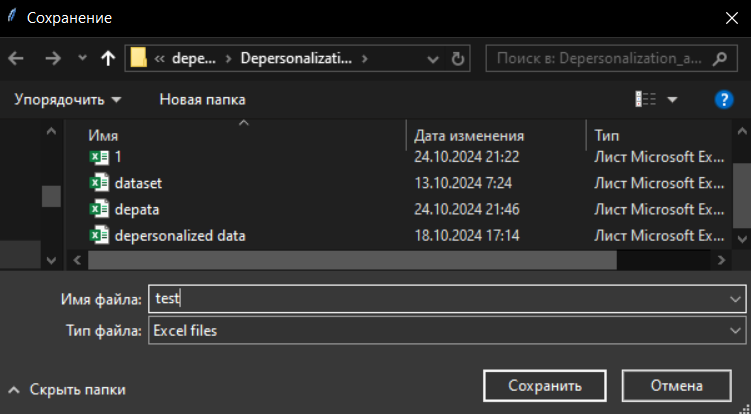
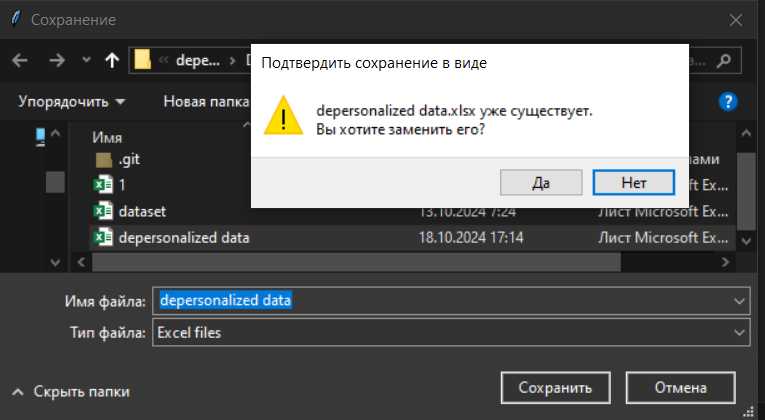
[**https://github.com/FraaaM/Depersonalization\_and\_anonymization\_of\_the\_data\_set**](https://github.com/FraaaM/Depersonalization_and_anonymization_of_the_data_set)

# Контрольный пример

* Запуск программы

Для запуска программы используйте exe файл Depersonalize in-store purchase data.exe или через Visual Studio Code (можно использовать другой IDE) откройте папку с программой и запустите main.py. Эта программа предназначена для **обезличивания** набора данных о покупках на основе заданных параметров (квази-идентификаторов).

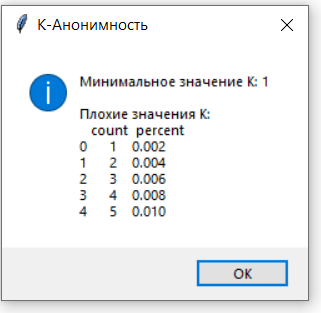
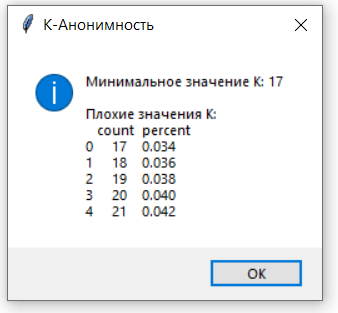
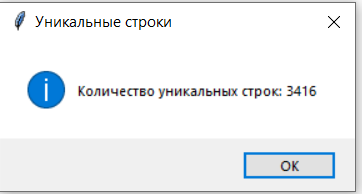
* Для работы программы загрузите файл для обезличивания. Необходимо нажать на кнопку Загрузить файл и выбрать нужный. Затем можно обезличить данные или вычислить k – анонимность . Если попытаться выполнить эти действия до загрузки файла, то появится предупреждение.
* Для обезличивания нужно выбрать квази-идентификаторы, если не выбрать ни один и подтвердить, то появится предупреждение

После успешного обезличивания, появится уведомление о том, что данные обезличены, затем нужно будет сохранить обезличенный датасет в excel файл создав новый(test) или перезаписав старый

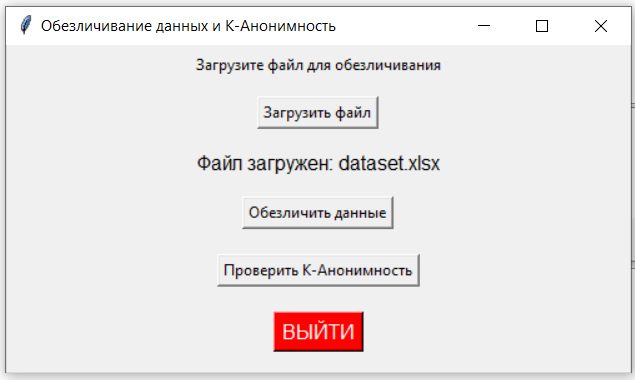
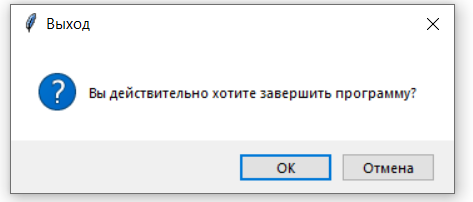
* Если загружен файл, то можно посчитать k – анонимность

Пример вывода если k > 1 и если k >= 1 (больше или равно)

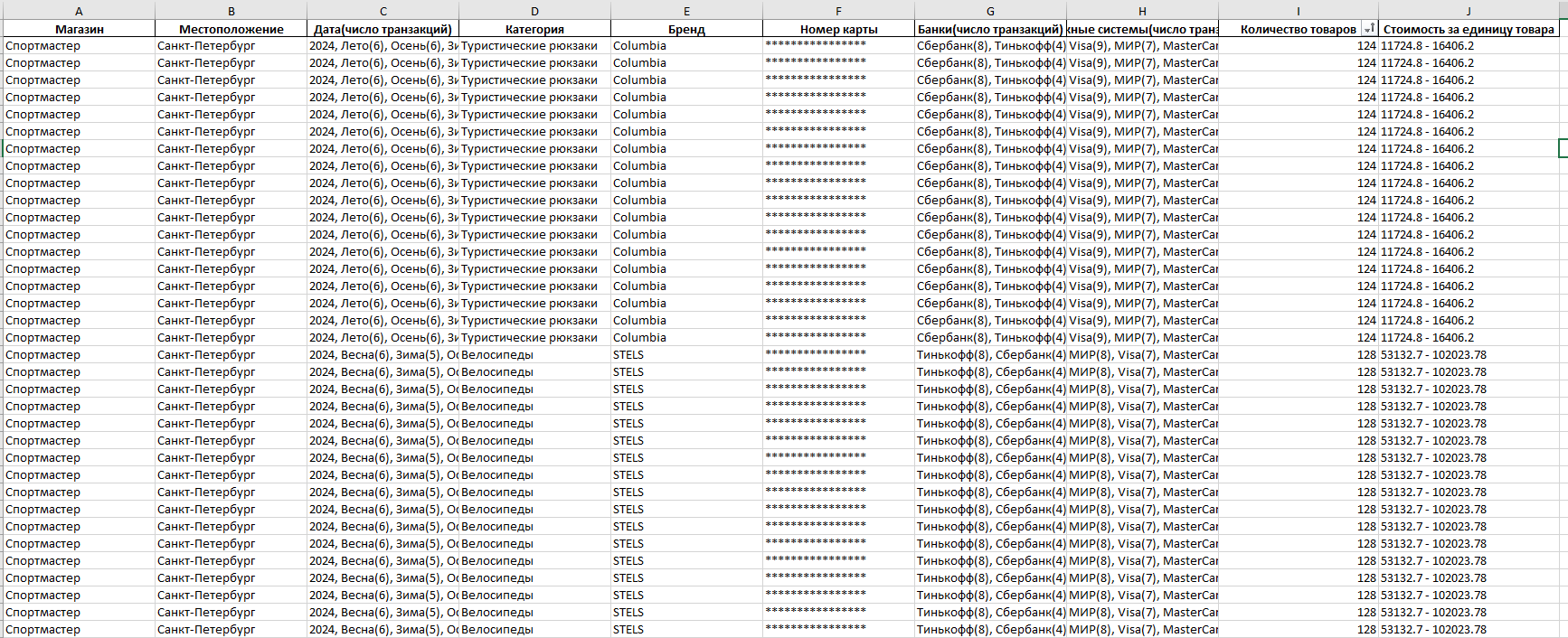
В первом случае выводится таблица с 5 минимальными значениями k – анонимности и рядом отображается процентное соотношение строк (записей) с данным значением k от общего числа строк (записей). Во втором случае к этому же выводится количество уникальных строк.

**k > 1** **k >= 1**

* Добавлена кнопка для завершения работы программы.



* После обезличивания, рекомендуется отфильтровать таблицу по столбцу Количество товаров по возрастанию, для более ясного восприятия.



# Вывод

В ходе выполнения данной работы была разработана программа, обеспечивающая обезличивание большого объема данных, моделирующего сценарии покупок в магазинах. Программа успешно реализует основные требования, включая:

* **Обезличивание датасета**: Программа эффективно обезличивает набор данных со значение k – анонимности большем 10 для 50.000 строк(записей), при этом не теряет необходимый объём информации, сохраняя полезность данных, что делает доступным их последующий анализ.
* **Выбор квази-идентификаторов**: Пользователи имеют возможность выбрать интересующие их данные для дальнейшего обезличивания и вычисления k - анонимности.
* **Разработан метод защити данных – k – анонимность**: Данный метод позволяет быстро определить степень защищённости информации и поддерживать нужный уровень безопасности.
* **Удобный экспорт данных**: Обезличенные данные сохраняются в формате Excel.
* **Добавлен графический интерфейс для работы с программой**.
* **Добавлен exe файл для быстрого и простого запуска программы**

# Источники

* Редактор блок-схем.

[*https://programforyou.ru/block-diagram-redactor*](https://programforyou.ru/block-diagram-redactor)

* tkinter — Библиотека для создания графических интерфейсов в Python.

[*https://docs.python.org/3/library/tkinter.html*](https://docs.python.org/3/library/tkinter.html%20)

* pandas – обработка и анализ данных*.*

*https://pandas.pydata.org/*

* Редактор блок-схем.

[*https://programforyou.ru/block-diagram-redactor*](https://programforyou.ru/block-diagram-redactor)

* pyinstaller – программа для создания exe файла: [*https://pyinstaller.org/en/stable/*](https://pyinstaller.org/en/stable/)
* *os* – библиотека, *предоставляющая функции для работы с операционной системой.*

*https://docs.python.org/3/library/os.html*