**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики-процессов управления**

**Программа бакалавриата**

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

**ОТЧЕТ**

**по лабораторной работе №1**

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

**на тему «Обезличивание данных»**

**Вариант – 3**

**Студент гр. 23Б15-пу**

**Крылов А. С.**

**Преподаватель**

**Дик А.Г.**

**Санкт-Петербург**

**2024 г.**

Содержание

[Цель работы 2](#_Toc179512760)

[Описание задачи (формализация задачи) 2](#_Toc179512761)

[Теоретическая часть 3](#_Toc179512762)

[Основные шаги программы 4](#_Toc179512763)

[Блок схема программы 5](#_Toc179512764)

[Описание программы 6](#_Toc179512765)

[Описание функций: 6](#_Toc179512766)

[Рекомендации пользователя 13](#_Toc179512774)

[Рекомендации программиста 13](#_Toc179512775)

[Исходный код программы 13](#_Toc179512776)

[Контрольный пример 14](#_Toc179512777)

# **Цель работы**

Целью лабораторной работы является разработка программы для обезличивания данных, чтобы сохранить конфиденциальность и при этом оставить данные полезными для анализа.

**Описание задачи**

Задача состоит в создании датасета для железнодорожных билетов со следующими требованиями:

1. **Чтение данных**:

Реализовать функцию для чтения входного набора данных (файл из первой лабораторной работы).

1. **Основные компоненты программы**:

Программа состоит из двух частей:

a. **Обезличивание данных**: Добавить несколько методов обезличивания.

b. **Расчёт K-анонимности**: Добавить функцию для расчёта K-анонимности.

1. **Указание квази-идентификаторов**:

Дать пользователю возможность выбирать, какие столбцы считать квази-идентификаторами.

1. **Методы обезличивания**:

Использовать методы обезличивания из: a. Локальное обобщение b. Агрегация c. Возмущение d. Микро-агрегация e. Перемешивание f. Создание псевдонимов g. Маскирование h. Локальное подавление i. Удаление атрибутов j. Декомпозиция

1. **Расчёт K-анонимности**:

Рассчитать K-анонимность для обезличенного набора данных.

1. **Низкие значения K-анонимности**:

Найти и показать пять самых низких значений K-анонимности (или все, если их меньше пяти) в процентах от всего набора.

1. **Анализ уникальных записей**:

Подсчитать количество уникальных строк по квази-идентификаторам. Если K=1, показать эти уникальные записи.

1. **Критерии K-анонимности**:

Определить приемлемые значения K-анонимности в зависимости от размера набора данных: a. До 51 000 записей: K ≥ 10. b. До 105 000 записей: K ≥ 7. c. До 260 000 записей: K ≥ 5.

1. **Оценка полезности данных**:

Сравнить обезличенные данные с исходными, чтобы проверить, остались ли они полезными для анализа.

# **Теоретическая часть**

Для создания программы по обезличиванию данных используются несколько модулей:

1. **interface.py**: Основной интерфейс программы, через который происходит взаимодействие пользователя с программой. Позволяет загружать набор данных, указывать квази-идентификаторы, выбирать методы обезличивания и рассчитывать K-анонимность.
2. **utils2.py**: Основные функции для обезличивания данных. Модуль содержит функции для локального обобщения, агрегации, маскирования и других методов обезличивания. Также реализованы функции для подсчета K-анонимности и оценки полезности данных.

# **Основные шаги программы**

1. Запуск программы (interface.py):
2. Пользователь вводит пути к файлам ввода и вывода датасетов, выбирает квази – идентификаторы.
3. В зависимости от выбора действия производится либо рассчет k\_anonimity, либо обезличивание исходного датасета.

# **Блок схема программы**

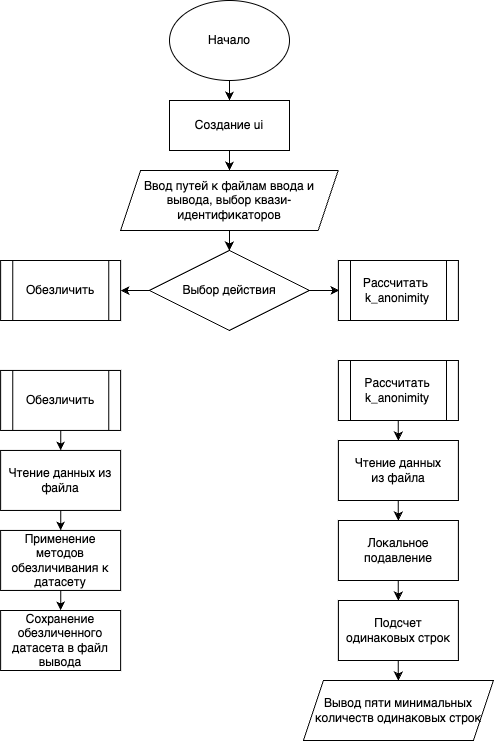


Рис 1. Блок-схема программы

# **Описание программы**

Программная реализация написана на языке Python 3.11.9 с использованием следующих библиотек: pandas, numpy, tkinter. Программа организована через модульную структуру с акцентом на эффективную работу с данными. Реализованы такие методы обезличивания как локальное подавление, локальное обобщение, маскеризация. Рандомизация и др. не применялись, так как в соответствии с целью работы были выбраны методы, обеспечивающие улучшение k\_anonimity.

## 

## Описание функций:

# **Из файла utils2.py:**

## Таблица 1. Функция `day`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметр | Тип | Описание |
| dt | datetime | Объект даты, из которого извлекается значение дня. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| int | День месяца из переданной даты. |

## Таблица 2. Функция `get\_class\_of\_route`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметр | Тип | Описание |
| x | float | Базовая стоимость маршрута, на основе которой определяется его класс. |

### Внутренние переменные

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Описание |
| pricing | dict | Словарь, связывающий классы маршрутов с их стоимостью. |
| key | str | Текущий класс маршрута для проверки. |
| value | int | Стоимость, соответствующая определенному классу маршрута. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| str | Класс маршрута, соответствующий указанной стоимости `x`. |

## Таблица 3. Функция `get\_train\_type`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметр | Тип | Описание |
| train\_number | str | Номер поезда, из которого извлекаются цифры для анализа. |

### Внутренние переменные

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Описание |
| train\_number | int | Номер поезда, преобразованный в целое число. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| str | Тип поезда в зависимости от его номера (например, "скорые поезда", "пассажирские поезда" и т.д.). |

## Таблица 4. Функция `get\_quasi\_identifiers`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметр | Тип | Описание |
| data | list of dict | Список записей данных, в которых необходимо выделить квази-идентификаторы. |
| quasi\_identifiers | list | Список квази-идентификаторов, которые необходимо выделить в каждой записи. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| list of dict | Список записей, каждая из которых содержит только квази-идентификаторы. |

## Таблица 5. Функция `compare\_datasets`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметр | Тип | Описание |
| original | list of dict | Оригинальный набор данных для сравнения. |
| anonymized | list of dict | Анонимизированный набор данных для сравнения. |
| quasi\_identifiers | list | Список квази-идентификаторов, которые используются для сравнения. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| tuple | Кортеж из трех значений: количество уникальных записей в оригинальном наборе, количество уникальных записей в анонимизированном наборе, процент сохраненных уникальных записей. |

## Таблица 6. Функция `calculate\_k\_anonymity`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметр | Тип | Описание |
| df | DataFrame | Набор данных, для которого вычисляется показатель `k-анонимности`. |
| column\_names | list | Список имен столбцов, по которым производится группировка данных. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| tuple | Кортеж, содержащий значение `k-анонимности` и список из 5 групп с наименьшей `k-анонимностью`. |

## Таблица 7. Функция `suppress`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметр | Тип | Описание |
| df | DataFrame | Набор данных, в котором необходимо подавить определенные значения. |
| quasi\_identifiers | list | Список квази-идентификаторов, которые необходимо подавить. |
| n | int | Максимальное количество записей, которые можно подавить. |

### Внутренние переменные

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Описание |
| df\_suppressed | DataFrame | Копия набора данных с применением подавления значений. |
| group\_counts | Series | Количество записей в каждой группе, определенной квази-идентификаторами. |
| group\_counts\_sorted | Series | Отсортированные группы данных по возрастанию количества записей. |
| suppressed\_count | int | Текущее количество подавленных записей. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| tuple | Кортеж, содержащий подавленный DataFrame и количество подавленных записей. |

## Таблица 8. Функция `open\_tickets`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметр | Тип | Описание |
| path | str | Путь к файлу Excel с данными о билетах. По умолчанию используется `'../data/Tickets.xlsx'`. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| DataFrame | Данные о билетах, загруженные из Excel файла. |

## Таблица 9. Функция `anonymise\_tickets`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметр | Тип | Описание |
| path | str | Путь к файлу Excel с данными о билетах. По умолчанию `'../data/Tickets.xlsx'`. |
| outpath | str | Путь к выходному файлу Excel с анонимизированными данными. По умолчанию `'../data/Tickets\_anon.xlsx'`. |

### Внутренние переменные

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Описание |
| df | DataFrame | Данные о билетах, загруженные из файла и подлежащие анонимизации. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| DataFrame | Анонимизированные данные о билетах. |

# **Из файла interface.py:**

## Таблица 10. Функция `show\_selected`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметры | Тип | Описание |
| - | - | Функция не принимает параметров. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| None | Выводит в консоль список выбранных элементов (состояние чекбоксов). |

## Таблица 11. Функция `get\_paths`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметры | Тип | Описание |
| - | - | Функция не принимает параметров. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| tuple | Возвращает кортеж из двух строк: путь к файлу и путь к выходному файлу. |

## Таблица 12. Функция `get\_checked\_items`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметры | Тип | Описание |
| - | - | Функция не принимает параметров. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| list | Возвращает список выбранных квази-идентификаторов (например, "ФИО", "Паспорт" и т.д.). |

## Таблица 13. Функция `set\_entries`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметры | Тип | Описание |
| values | list | Список значений, которые необходимо вставить в соответствующие поля ввода. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| int | Возвращает 0 после успешного выполнения. |

## Таблица 14. Функция `anon`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметры | Тип | Описание |
| - | - | Функция не принимает параметров. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| DataFrame | Возвращает DataFrame с анонимизированными данными о билетах и сохраняет его в указанный файл. |

## Таблица 15. Функция `open`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметры | Тип | Описание |
| - | - | Функция не принимает параметров. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| DataFrame | Возвращает DataFrame с данными о билетах из указанного входного файла. |

## Таблица 16. Функция `open\_anon`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметры | Тип | Описание |
| - | - | Функция не принимает параметров. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| DataFrame | Возвращает DataFrame с анонимизированными данными из указанного выходного файла. |

## Таблица 17. Функция `calculation`

### Параметры

|  |  |  |
| --- | --- | --- |
| Параметры | Тип | Описание |
| - | - | Функция не принимает параметров. |

### Возвращаемое значение

|  |  |
| --- | --- |
| Тип | Описание |
| None | Выполняет расчет и отображает результат в соответствующих полях интерфейса. |

**Рекомендации пользователя**

* Для запуска программы, убедитесь, что у вас установлен Python и необходимые библиотеки. Запустите код в среде разработки или командной строке и используйте интерфейс для настройки параметров и генерации данных.
* Убедитесь, что все модули программы находятся в одной директории для корректного выполнения.
* Запуск производится файлом main.py, который автоматически генерирует расписание в файл timetable.xlsx если его нет.
* Периодически проверяйте корректность данных перед генерацией билетов.
* При желании использовать свой файл расписания, убедитесь, что он называется traisbook.xlsx и его структура его заголовков совпадает с используемой: Номер поезда, категория, город отправления, город назначения, время следования (часы).
* Настройте веса для платежных систем и банков в соответствии с вашими требованиями, но убедитесь, что веса банков и платежных систем больше нуля.

# **Рекомендации программиста**

* Поддерживайте актуальность используемых библиотек и версии Python для сохранения актуальности и работоспособности кода на современных системах.
* Организуйте код в соответствии с best practices, следите за четким именованием переменных и функций.
* Периодически проводите тестирование на различных входных данных для обеспечения надежности и корректности программы.

# **Исходный код программы**

[**https://github.com/akryloff/spbu-algorithms-and-data-structures**](https://github.com/akryloff/spbu-algorithms-and-data-structures)

# **Контрольный пример**

Запустите код в среде разработки или командной строке. На экране появится интерфейс программы (Рис. 2)

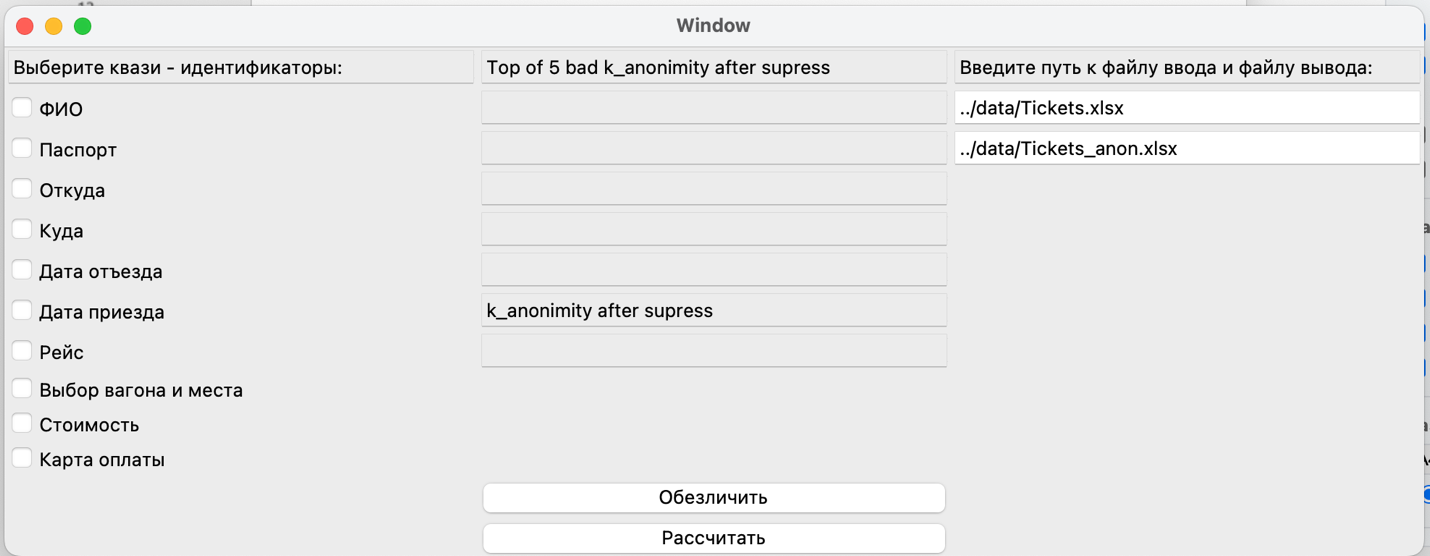


Рис 2. Интерфейс программы

Введите пути к файлам ввода и вывода (можно оставить значения по умолчанию), выберите желаемые квази-идентификаторы, нажмите одну из кнопок – “Обезличить” для изменения датасета методами обезличивания, или “Рассчитать” для вывода плохих k и конечного k.

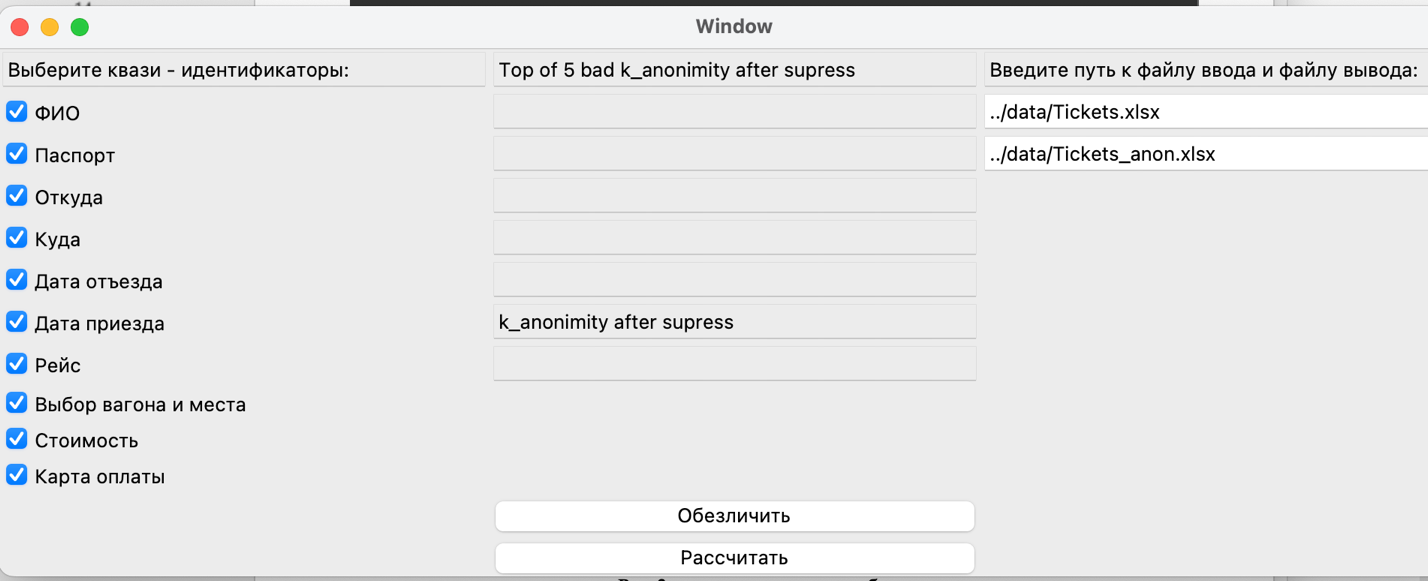
****

Рис 3. пример выбора идентификаторов (выбраны все)

Рассчитаем k для необезличенного датасета. Получим вполне ожидаемый результат – 1, так как необезличенный датасет содержит много уникальных строк. 

Рис 4. Результат на оригинальном датасете

Теперь нажмем кнопку “Обезличить”, дождемся загрузки и нажмем “Рассчитать”



Рис 5. Результат после обезличивания

Получаем результат удовлетворяющий оптимальным значениям. Для датасета порядка 150000 строк, k\_anonimity = 9.

**Вывод**

В рамках данной работы были исследованы методы обезличивания. Разработан алгоритм, который учитывает особенности пассажирских данных, маршрутов, типов поездов и вагонов, а также ценообразования билетов для эффективного обезличивания данных такого типа. Было реализовано программное обеспечение для автоматического обезличивания датасета, включающего такие данные, как личные данные пассажиров, информация о поездах, билетах и платежных системах. Программа позволяет выбирать квази-идентификаторы для вычисления k\_anonimity в соответствии с необходимыми данными.