**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики-процессов управления**

**Программа бакалавриата**

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

**ОТЧЁТ**

**По лабораторной работе № 1**

**По дисциплине “Алгоритмы и структуры данных”**

**На тему “ Создание датасета ”**

**Вариант 2.**

**Студент гр. 24.Б16-пу**

**Музиров Б.Р.**

**Преподаватель**

**Дик А.Г.**

**Санкт-Петербург**

**2025 г.**

Оглавление

[**1.** **Цель работы** 3](#_Toc209554251)

[**2.** **Описание задачи** 3](#_Toc209554252)

[**3.** **Теоретическая часть** 3](#_Toc209554253)

[*3.1. Что-то зачем-то* 3](#_Toc209554254)

[**4.** **Основные шаги программы** 3](#_Toc209554255)

[**5.** **Блок схема программы** 3](#_Toc209554256)

[**6.** **Описание программы** 4](#_Toc209554257)

[**7.** **Рекомендации пользователя** 5](#_Toc209554258)

[**8.** **Рекомендации программиста** 5](#_Toc209554259)

[**9.** **Контрольный пример** 6](#_Toc209554260)

[**10.** **Исследование** 6](#_Toc209554261)

[**Вывод** 6](#_Toc209554262)

1. **Цель работы**

Цель работы — разработать программу для генерации синтетического датасета посетителей платной поликлиники, имитирующего реальные данные.

1. **Описание задачи**

Задача состоит в создании датасета, обладающего следующей структурой:

1. **ФИО:** только славянские варианты.
2. **Паспортные данные:** только русские, белорусские и казахские паспорта.
3. **СНИЛС:** уникальный, но привязан к клиенту, может повторяться при повторном посещении.
4. **Симптомы:** выборка минимум из 5000 симптомов, может быть комбинация итоговых симптомов (не более 10 штук).
5. **Выбор врача:** выборка минимум из 50 врачей.
6. **Дата посещения врача:** в рабочие время и дни недели, повторное посещение минимум через 24 часа после получения анализов.
7. **Анализы:** выборка минимум из 250 анализов, может быть комбинация итоговых анализов (не более 5 штук).
8. **Дата получения анализов:** в рабочие время и дни (через 24-72 часа).
9. **Стоимость анализов:** только в рублях.
10. **Карта оплаты:** максимальное количество повторов - 5 раз.
11. **Количество строк датасета**: минимум 50000.
12. **Теоретическая часть**

Для создания датасета использован программный модуль и набор csv таблиц:

1. functions.py: Файл, содержащий функции для генерации всех необходимых полей медицинской карточки пациента
2. main.py: Файл, содержащий функции работы с таблицами формата .csv, генерации готовой медицинской карты, записи медицинских карт в таблицу формата .csv. Данные для генерации берутся из файлов names.csv, medicine.csv, costs.csv, находящихся в директории data. Итог работы программы — файл output.csv
3. names.csv — таблица, которая содержит данные для генерации ФИО посетителей поликлиники.
4. medicine.csv — таблица, которая содержит данные для генерации симптомов, анализов, которые требуются врачу для выявления данного симптома, а также соответствующих типов врачей.
5. costs.csv — таблица, которая содержит данные о ценах для каждого анализа.

Кроме того, при организации процесса генерации следует помнить, что:

* Количество строк в датасете определяется вводом пользователя и начинается с 50000.
* Дата получения анализов должна зависеть от даты посещения врача (быть в диапазоне 24-72 часа после)
* Поля должны приводиться к строковому типу данных, чтобы избежать их некорректного отображения в таблице
* Веса банков и платежных систем определяются пользователем и должны в сумме давать положительное число
* Оплата не может производиться с одной и той де карты более 5 раз

1. **Основные шаги программы**

**4.1. Загрузка списка имён из файла**  
Из файла names.csv с помощью модуля csv считываются мужские и женские фамилии, имена и отчества. Данные сохраняются в два отдельных списка (MALE, FEMALE).

**4.2. Генерация персональных данных**

* ФИО формируется случайным образом из загруженных списков (функция generate\_fullname).
* Паспортные данные генерируются с использованием случайного выбора серии и номера (generate\_passport).
* СНИЛС формируется по алгоритму с контрольной суммой (generate\_snils).
* Дата посещения врача и получения анализов вычисляется случайным образом в пределах рабочего времени, с учётом будних дней (generate\_datetime, get\_back\_analysis).
* Генерация номера банковской карты с учётом весов вероятности выбора банка и проверкой по алгоритму Луна (generate\_card\_number).

**4.3. Загрузка справочников**  
Из файлов medicine.csv и costs.csv подгружаются:

* список симптомов, соответствующих врачей и анализов;
* стоимость анализов.

**4.4. Формирование медицинской карты пациента**  
В функции make\_medcard:

* выбирается случайное количество симптомов и анализов, связанных с ними;
* определяется соответствующий врач;
* для каждого анализа находится стоимость;
* формируется запись со всеми полями: ФИО, Паспортные данные, СНИЛС, Симптомы, Выбор врача, Дата посещения врача, Анализы, Дата получения анализов, Стоимость анализов, Карта оплаты.

**4.5. Генерация набора медицинских карт**  
Функция generate\_output\_csv многократно вызывает make\_medcard, пока не будет достигнуто заданное количество строк (например, 50 000).

**4.6. Сохранение результата в файл**  
Все сгенерированные медицинские записи выгружаются в файл output.csv с заголовками и разделителем ; (функция output).

1. **Блок схема программы**



Рис 1. Блок-схема основной программы

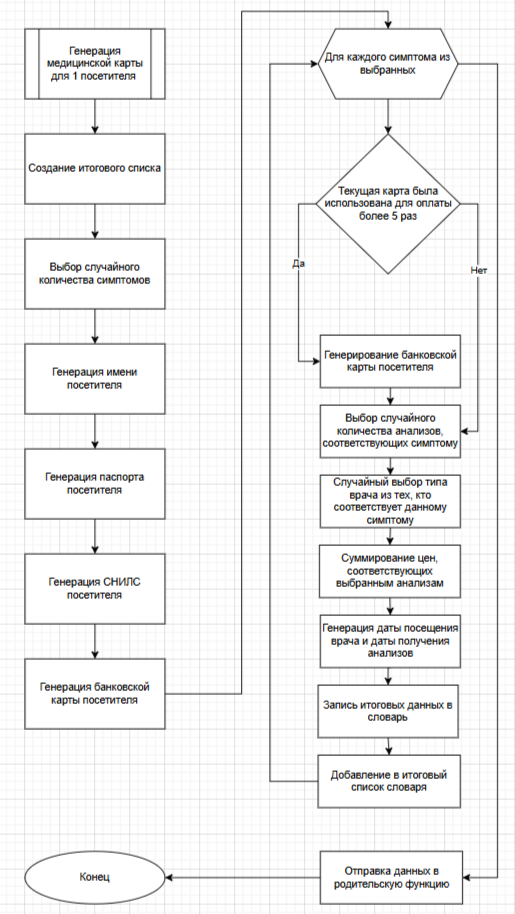


Рис 2. Блок-схема подпрограммы.

1. **Описание программы**

Програмная реализация задачи написана на языке Python с использованием модулей random, csv, datetime. Программа разбита на 2 файла, functions.py и main.py, где первый содержит функции, связанные с генерацией отдельных полей медицинской карты посетителя, а второй — функции, связанные с формированием целостной медицинской карты посетителя, чтением и записью файлов в формате csv, а также генерацию итогового файла.

Таблица 1. Описание функций из файла functions.py

|  |  |  |
| --- | --- | --- |
| Функция | Тип возвращаемого значения | Описание функции |
| load\_names(source) | tuple | Загружает список имён из CSV-файла и разделяет их на мужские и женские. |
| generate\_fullname(gender, Male, Female) | str | Формирует случайное ФИО (фамилия, имя, отчество) на основе полов. |
| generate\_snils() | str | Генерирует случайный номер СНИЛС (9 цифр + контрольное число). |
| generate\_passport() | str | Создаёт случайные паспортные данные РФ. |
| generate\_datetime(time\_begin="08:00", time\_end="21:00") | str | Генерирует случайную дату и время приёма врача в будний день. |
| get\_back\_analysis(date\_time, time\_begin="08:00", time\_end="21:00") | str | Вычисляет дату и время получения анализов спустя 1–3 дня после визита к врачу. |
| calculate\_luhn\_checksum(partial\_card\_number) | str | Вычисляет контрольную цифру для номера банковской карты по алгоритму Луна. |
| generate\_card\_number(banks) | str | Генерирует полный номер банковской карты (16 цифр). |

Таблица 2. Описание функций из файла main.py

|  |  |  |
| --- | --- | --- |
| Функция | Тип возвращаемого значения | Описание функций |
| make\_medcard(bank\_card\_data, male\_list, female\_list) | list | Формирует набор медицинских записей (медкарту пациента) на основе симптомов, анализов, врачей и личных данных. |
| get\_csv(source) | list | Загружает данные из CSV-файла в список. |
| output(source, data) | None | Сохраняет список медицинских записей в CSV-файл. |
| generate\_output\_csv(source, bank\_card\_data, male\_list, female\_list, limit) | None | Формирует итоговый CSV-файл с заданным количеством записей. |

# **Рекомендации пользователя**

Для корректного выполнения программы следуйте приведённым шагам:

1. Клонируйте или скачайте проект.
2. Запустите программу с помощью команды (находитесь в той же директории, что и программа): python main.py

Программа поддерживает 2 аргумента командной строки:

Таблица 3. Описание аргументов для консольного запуска программы

|  |  |  |
| --- | --- | --- |
| Параметр | Описание | По умолчанию |
| -n, --num | Количество строк в выходном файле | 50000 |
| -w, --weights | Веса для банков (Сбербанк, Тинькофф, ВТБ, Альфа, Газпромбанк) | 0.3 0.2 0.2 0.2 0.1 |

Пример: python main.py -n 200000 -w 0.5 0.1 0.1 0.1 0.2

1. Готовый датасет будет сохранён в папку проекта

# **Рекомендации программиста**

Для внесения изменений в исходный код программы необходимо наличие устройства на базе операционных систем Linux, macOS или Windows, а также среды разработки, поддерживающей запуск python версии 3.8 и выше

Исходный код доступен по ссылке:

<https://github.com/>

# **Контрольный пример**

Для проверки корректности работы программы проведём запуск с параметрами по умолчанию.

**Шаг 1.** Запуск программы

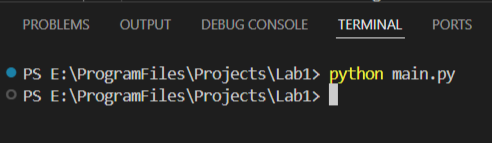


Рис.3 Скриншот запуска

**Шаг 2.** Проверка вывода в консоль  
При запуске программа автоматически подгружает необходимые данные из файлов names.csv, medicine.csv и costs.csv, а также применяет веса банков по умолчанию:

0.3 0.2 0.2 0.2 0.1

что соответствует:

* Сбербанк — 30%
* Тинькофф — 20%
* ВТБ — 20%
* АльфаБанк — 20%
* ГазпромБанк — 10%

**Шаг 3.** Проверка результата  
После завершения генерации в корневой директории проекта появляется файл output.csv В нём содержится 50 000 записей с полями: ФИО; Паспортные данные; СНИЛС; Симптомы; Выбор врача; Дата посещения врача; Анализы; Дата получения анализов; Стоимость анализов; Карта оплаты

Пример первой строки файла:

Семенов Назар Савельевич;"=""6168943484""";775-566-500 51;Учащённое сердцебиение;Кардиолог;2025-04-11T09:28;Ферритин | Тироксин свободный (Т4 свободный);2025-04-14T08:00;1700;"=""2202205187658718"""

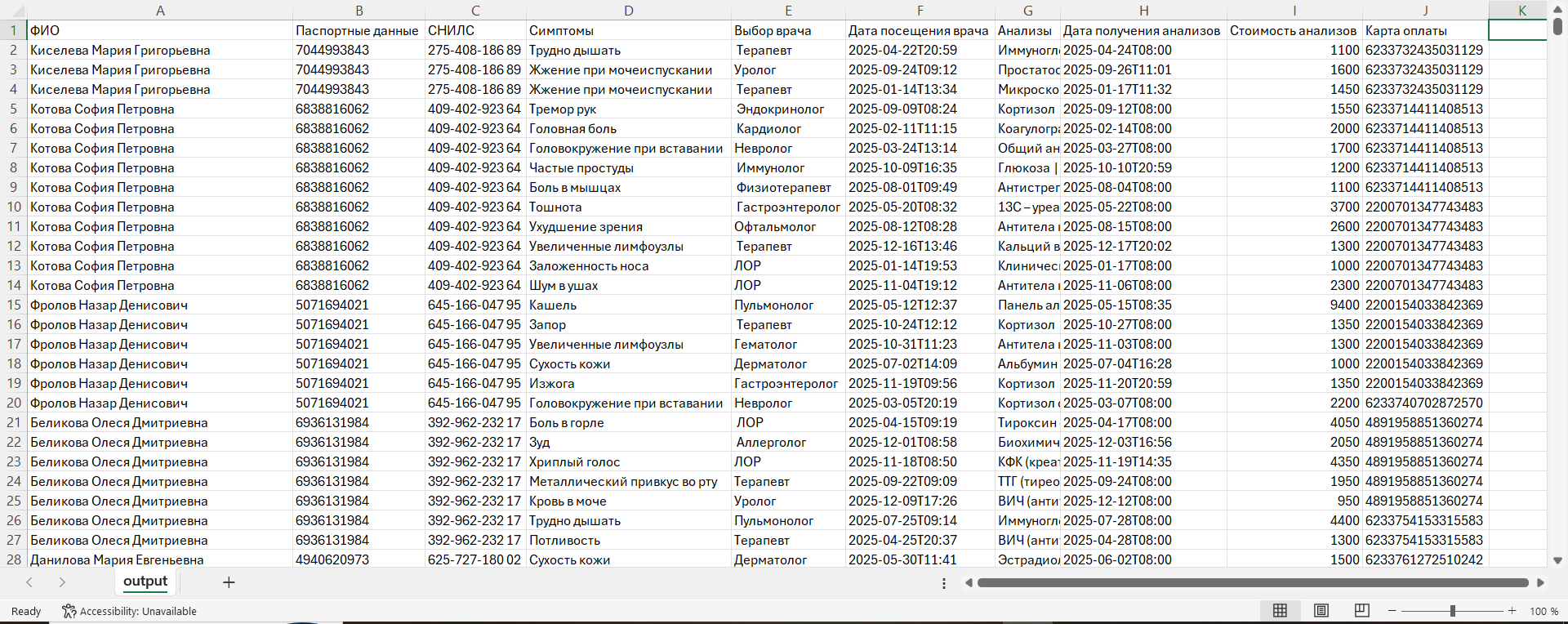


Рис.4 Скриншот итогового датасета

**Шаг 4.** Проверка соответствия условий

* Количество строк = 50 000
* Повторы карты не более 5 раз
* Даты анализов следуют за датами приёмов
* СНИЛС и паспортные данные соответствуют формату

# **Исследование**

Описываем свои изыскания по проделанной работе, если есть исследование.

# **Вывод**

Обычно подводим общий вывод по работе проделанной.

Н: В рамках данной работы были изучены и реализованы ххххх

Также было написано программное обеспечение с графическим интерфейсом для применения ххххх:

В итоге исследования было выявлено, что ххххх