**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**Факультет прикладной математики – процессов управления**

**отчет**

**по лабораторной работе**

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

**на тему «Генерация датасета»**

**2 вариант**

|  |  |  |
| --- | --- | --- |
| Студент гр. 23.Б16 |  | Барафанов А.А. |
| Преподаватель |  | Дик А.Г. |

**Санкт-Петербург**

**2024 г.**

**Оглавление**

[**1. Цель работы** 3](#_Toc178330770)

[**2. Задача (формализация)** 3](#_Toc178330771)

[**3. Теоретическая часть** 3](#_Toc178330772)

[**4. Описание алгоритма** 4](#_Toc178330773)

[**5. Описание программы** 7](#_Toc178330774)

[**5.1 Описание классов** 7](#_Toc178330775)

[**5.2 Описание функций** 9](#_Toc178330776)

[**6. Рекомендации пользователю** 10](#_Toc178330777)

[**7. Рекомендации программисту** 10](#_Toc178330778)

[**8. Контрольный пример** 11](#_Toc178330779)

[**9. Вывод** 12](#_Toc178330780)

# **Цель работы**

Написать алгоритм, генерирующий синтетические данные для платной поликлиники.

# **Задача (формализация)**

Сгенерировать датасет со следующим набором свойств:

1. *ФИО:* (пример вывода - Иванов Иван Иванович)
2. *Паспортные данные:* уникальные значения
3. *СНИЛС:* уникальные значения
4. *Симптомы:* (пример вывода - боль в горле)
5. *Выбор врача:* (пример вывода - лор)
6. *Дата посещения врача:* (пример вывода - 2020-01-22T08:30+03:00)
7. *Анализы:* (пример вывода - мазок на ковид)
8. *Дата получения анализов:* (пример вывода - 2020-01-24T09:30+03:00)
9. *Стоимость анализов:* стоимость в рублях
10. *Карта оплаты:* генерация с существующими бины банков и платежных систем

# **Теоретическая часть**

*Описание создания датасета:*

Формирование словарей с возможными именами, фамилиями, отчествами, врачами, анализами и симптомами и последующая генерация пациентов на основе этих данных. Определенный вариант генерации даты и времени. Последующий вывод данных в файл формата xml или csv.

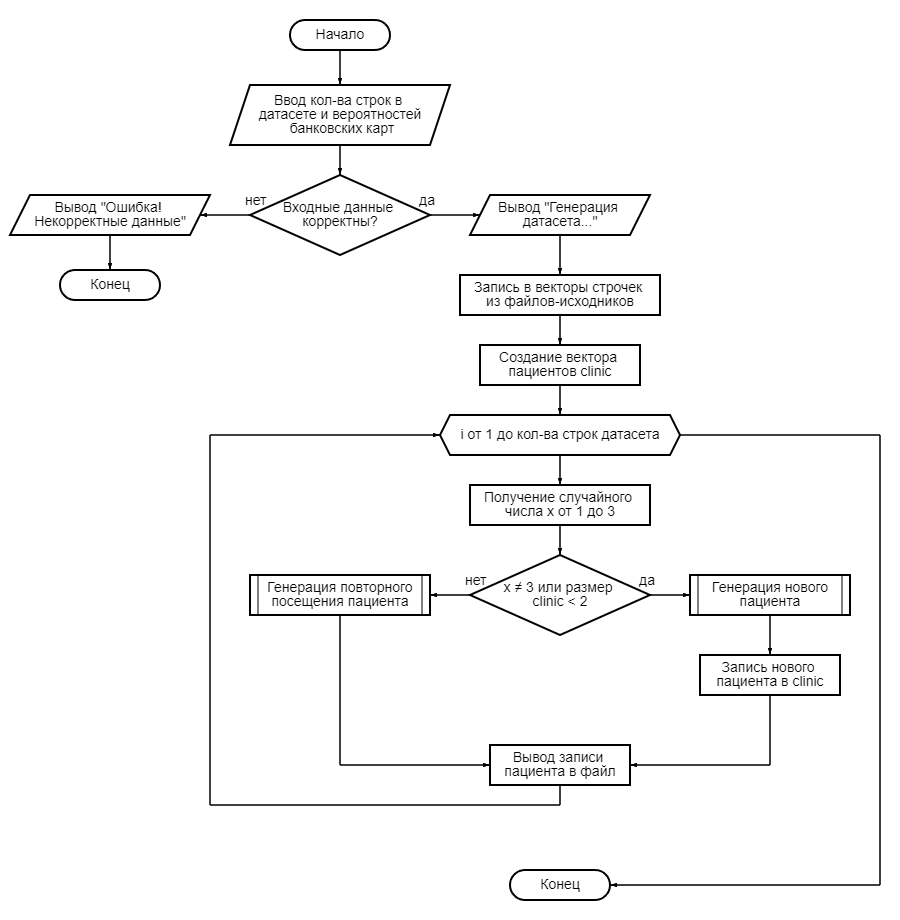
*Ограничения датасета:*

* Строк в датасете – минимум 50 тысяч;
* Только славянские ФИО;
* Только белорусские, русские и казахские паспорта;
* Размеры словарей: симптомы – min 5 тыс., анализы – min 250 штук., врачи – min 50 специальностей;
* Определенные временные периоды между повторными посещениями и получением анализов;
* Ограничения на оплату с одной карты (не более 5 раз).

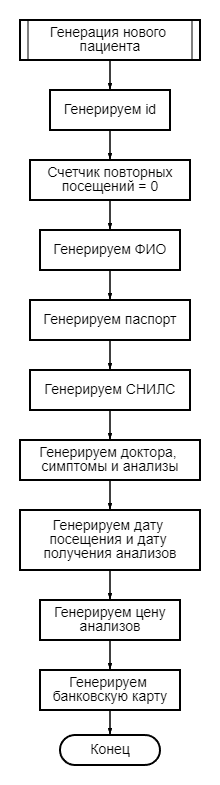
# **Описание алгоритма**

1. Считывание данных, введенных пользователем (количество строк в датасете, вероятность банковских карт);
2. Проверка данных на корректность;
3. Считывание файлов с необходимыми словарями в вектора: *male\_names\_vector, male\_surnames\_vector, male\_midnames\_vector, female\_names\_vector, female\_surnames\_vector, female\_midnames\_vector, symptoms\_vector, doctors\_vector, analyzes\_vector;*
4. Запуск цикла, с определенной вероятностью генерации нового пациента, либо повторного посещения;
5. Последовательное заполнение полей для каждого пациента: ФИО, паспорт, СНИЛС, врач, симптомы, анализы, дата посещения и дата получения анализов, цена анализов, банковская карта посредством методов: *Generate\_fio, Generate\_passport, Generate\_snils, Generate\_doctor\_symp\_an, Generate\_date\_vis\_and\_an, Generate\_price, Generate\_card;*
6. Запись пациента в файл outout.csv.
7. Закрытие файла вывода.

*Блок-схемы, иллюстрирующие шаги алгоритма показаны на рисунках 4.1-4.2.*



*Рисунок 4.1. Блок-схема основной программы.*



*Рисунок 4.2. Блок-схема подпрограмм.*

# **Описание программы**

## **Описание классов**

В программе используется 1 класс. В таблице 5.1 представлено его описание.

*Таблица 5.1. Описание классов*

|  |  |  |
| --- | --- | --- |
| Имя класса | Наследование | Описание класса |
| Person | — | Выполнение алгоритма генерации пациента |

Описание переменных класса Person представлено в таблице 5.2.

*Таблица 5.2. Описание переменных класса Person*

|  |  |  |  |
| --- | --- | --- | --- |
| Тип | Права доступа | Имя | Описание |
| static int | public | id\_generate | Статическая переменная, созданная для генерации идентификаторов |
| int | public | count\_povtor\_visit | Кол-во повторных посещений |
| string | private | id | Идентификатор пациента |
| struct tm | private | timeStruct\_vis | Дата посещения |
| struct tm | private | timeStruct\_an | Дата получения анализов |
| string | private | fio | ФИО пациента |
| string | private | passport | Паспорт |
| string | private | snils | СНИЛС |
| string | private | symptoms | Симптомы пациента |
| string | private | doctor | Лечащий врач |
| string | private | analyzes | Список анализов пациента |
| string | private | price | Цена анализов |
| string | private | card | Банковская карта платежа |

В таблице 5.3 представлено описание методов класса Person.

*Таблица 5.3. Описание методов класса Person*

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Модификатор  доступа/ Возвращаемый тип | Входные данные | Описание |
| SetId | public  void | - | Присваивание идентификатора пациенту |
| PrintPerson | public  void | Файл вывода | Вывод записи о пациенте в файл вывода |
| Generate\_fio | public  void | Векторы, хранящие мужские и женские имена, фамилии и отчества | Генерация ФИО пациента |
| Generate\_passport | public  void | - | Генерация паспорта пациента |
| Generate\_snils | public  void | - | Генерация СНИЛС пациента |
| Generate\_doctor\_symp\_an | public  void | Векторы, хранящие врачей, симптомы и анализы | Генерация врача, проводившего прием, симптомов пациента и назначенных анализов |
| Generate\_date\_vis\_and\_an | public  void | - | Генерация даты посещения |
| Generate\_new\_vis\_and\_an | public  void | - | Генерация повторной даты посещения (если оно было) |
| Generate\_price | public  void | - | Генерация цены анализов |
| Generate\_card | public  void | Два числа: вероятности двух платежных систем | Генерация банковской карты |
| Generate\_Person | public  void | Векторы, хранящие мужские и женские имена, фамилии и отчества;  Векторы, хранящие врачей, симптомы и анализы;  Два числа: вероятности двух платежных систем | Генерация всей информации о пациенте |
| Repeat\_Person | public  void | Вектор, хранящий всех пациентов клиники;  Индекс выбранного пациента (который придет на повторный прием);  Векторы, хранящие врачей, симптомы и анализы;  Два числа: вероятности двух платежных систем | Генерация повторного посещения пациента |

## **5.2 Описание функций**

В программе используется 2 функции, выполняющие вспомогательные процедуры программы. В таблице 5.4 представлено их описание.

*Таблица 5.4. Описание функций*

|  |  |  |
| --- | --- | --- |
| Имя функции | Входные данные | Описание функции |
| GetRandomInt | Максимальное и минимальное число | Получение случайного числа в указанном диапазоне |
| GetVectorFromFile | Файл, из которого берутся данные;  Вектор, в который записываются данные | Запись данных из файла в вектор (по строкам) |

# **Рекомендации пользователю**

При вводе количества строк датасета учитывайте, что *количество строк в датасете* – целое положительное число, большее 50000.

Вводите вероятности всего *двух* платежных систем (третья будет рассчитана автоматически). Формат ввода – целое число от 1 до 100. Следите за тем, чтобы сумма двух чисел не превышала 100.

Убедитесь, что файлы с исходными данными находятся в одной директории с проектом.

Датасет будет сохранен в файл outout.csv в директории проекта. Если у вас возникают неполадки с кодировкой при открытии файла csv в Excel, настройте кодировку utf-8 в Excel. Для этого зайдите в *Данные->Получить данные->Из файла->Из текстового/csv файла*. Выберите файл outout.csv нажмите *Импорт* и настройте кодировку *utf-8*. Разделительный знак по умолчанию «;».

1. **Рекомендации программисту**

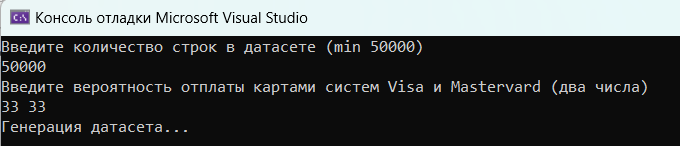
Для работы с программой должен быть установлен компилятор языка С++, а также сборщик проектов (CMake) или же интегрированная среда разработки (Visual Studio). Старайтесь поддерживать актуальные версии компиляторов и сред разработки. Поддерживайте актуальный стандарт языка (необходим стандарт не ниже С++11). Для запуска программы необходима 64-битная операционная система Windows.

1. **Контрольный пример**

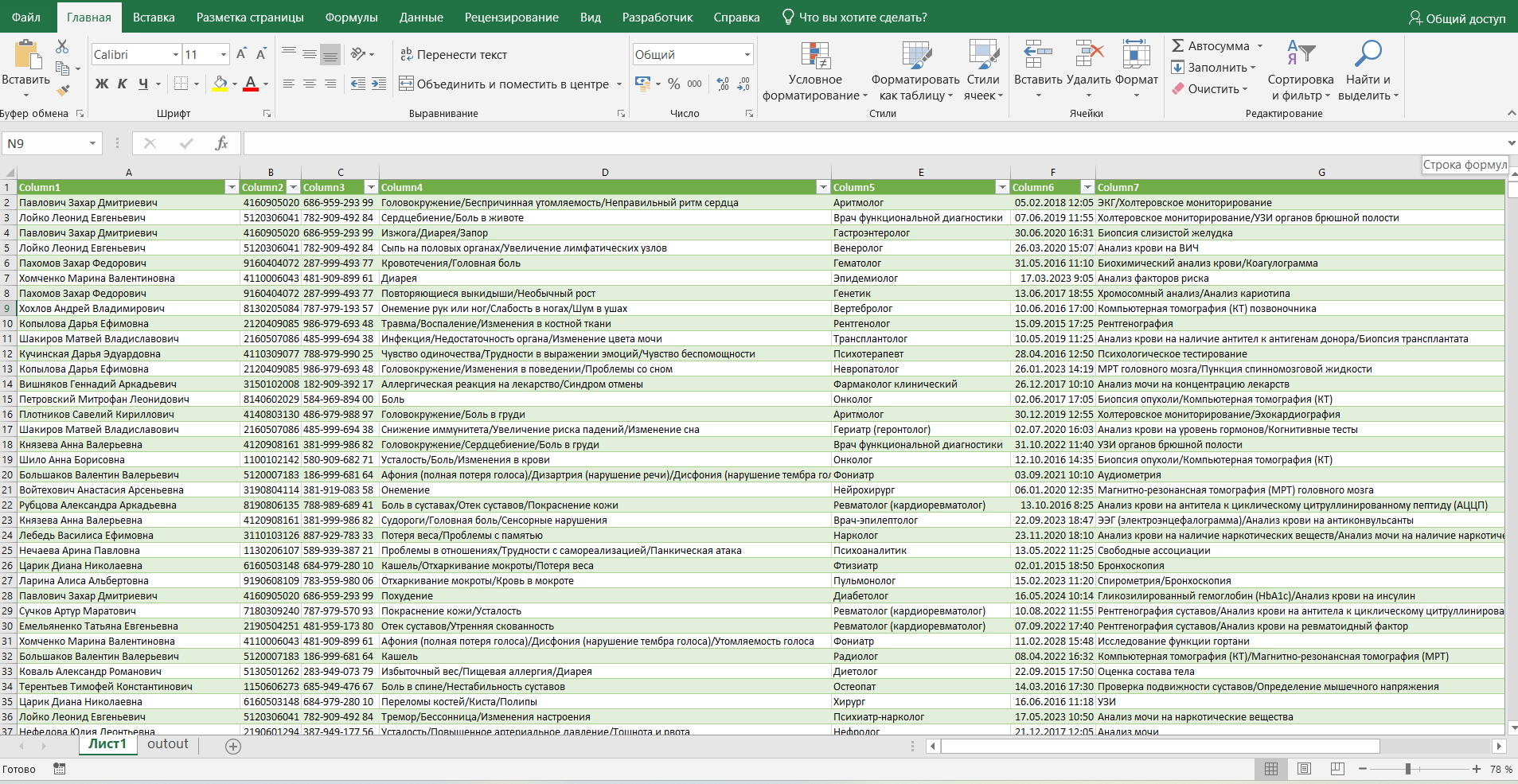
В этом разделе продемонстрированы два контрольных примера, показывающих ввод различных данных пользователем и полученные результаты.

Вероятности вводятся в консоль через пробел или через нажатие клавиши *Enter*. Если все числа введены корректно, то появится надпись «Генерация датасета…». В противном случае («Ошибка! Некорректные данные») требуется перезапуск программы и ввод корректных чисел. Пример ввода корректных входных данных показан на рисунке 8.1.

*Пример получившегося датасета показан на рисунке 8.2.*

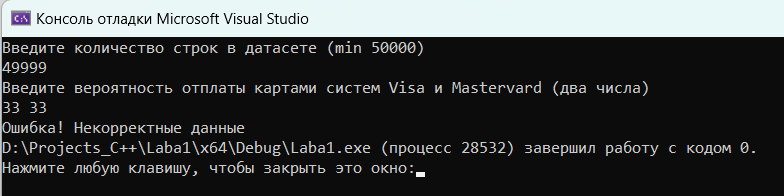


*Рисунок 8.1 Ввод данных в консоль*

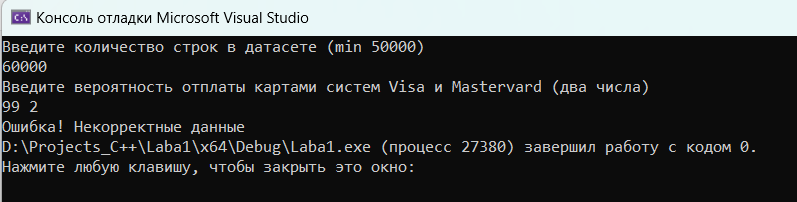


*Рисунок 8.2 Сгенерированный датасет*

*Пример ввода некорректных данных представлен на рисунках 8.3-8.4.*



*Рисунок 8.3 Некорректное число строк*



*Рисунок 8.4 Некорректные вероятности банковских карт*

1. **Вывод**

В ходе работы был построен алгоритм генерации синтетических данных на примере датасета для платной поликлиники, включающий многие атрибуты, такие как: ФИО, паспорт, доктор, анализы, банковская карта (и другие) и учитывающий, связанные с ними ограничения. Были приобретены знания в области генерации данных и работы с файлами, а также получена таблица с данными пациентов, которую можно использовать в будущих учебных проектах.