

Содержание

[**Введение** 4](#_Toc140335174)

* [**Цель практики** 4](#_Toc140335175)
* [**Задачи практики** 4](#_Toc140335176)

[**Выбор платформы и библиотек** 6](#_Toc140335177)

* [**Обоснование выбора платформы интерпретатора языка Python** 6](#_Toc140335178)
* [**Выбор необходимых библиотек** 7](#_Toc140335179)
* [**Вывод по разделу** 8](#_Toc140335180)

[**Простейший анализ данных** 12](#_Toc140335181)

* [**Получение данных. Создание списка со случайных чисел** 12](#_Toc140335182)
* [**Определение элементарных числовых характеристик** 13](#_Toc140335183)
* [**Визуализирование данных с помощью стандартных библиотек** 15](#_Toc140335184)
* [**Выводы по разделу** 18](#_Toc140335185)

[**Работа с библиотекой Numpy** 19](#_Toc140335186)

* [**Генерация массивов данных** 19](#_Toc140335187)
* [**Нахождение определителя А** 19](#_Toc140335188)
* [**Генерация Вектор-Столбца** 21](#_Toc140335189)
* [**Решение матричного уравнения** 21](#_Toc140335190)
* [**Рассчёт средней арифметической** 23](#_Toc140335191)
* [**Рассчёт суммы числе по каждой строке** 24](#_Toc140335192)
* [**Выводы по разделу** 25](#_Toc140335193)

[**Количественный анализ статистики продаж маркетплейса** 25](#_Toc140335194)

* [**Получение данных Dataset** 25](#_Toc140335195)
* [**Анализ датасетов** 26](#_Toc140335196)
* [**Определение числа товаров** 28](#_Toc140335197)
* [**Определение максимальной стоимости товара по каждой товарной группе** 29](#_Toc140335198)
* [**Рассчёт процента суммы покупок в штуках по категории для доставленных заказов, а также разбивку по штатам** 31](#_Toc140335199)
* [**Визуализируйте зависимость между средним число покупок и количеством символов в описании товара(шаг 25)** 33](#_Toc140335200)
* [**Визуализируйте зависимость между средним числом покупок и числом символов в названии(шаг 10)** 34](#_Toc140335201)
* [**Определение топ-5 продавцов, чаще всего отсылающим свою посылку в другие регионы** 35](#_Toc140335202)
* [**Выводы по разделу** 36](#_Toc140335203)

[**Список использованной литературы** 37](#_Toc140335204)

[**Приложение** 37](#_Toc140335205)

* [**Листинг программного кода для решения задачи индивидуального задания «Простейший анализ данных»** 37](#_Toc140335206)
* [**Листинг программного кода для решения задачи индивидуального задания «Работа с библиотекой Numpy»** 37](#_Toc140335207)
* [**Листинг программного кода для решения задачи индивидуального задания «Количественный анализ статистики продаж маркет-плейса»** 37](#_Toc140335208)

# **Введение**

Учебная практика (ознакомительная практика) является частью базового учебного плана основной профессиональной образовательной программы по направлению подготовки 09.03.03 Прикладная информатика и является видом учебной деятельности, направленной на формирование, закрепление, развитие практических навыков и компетенции в процессе выполнения определенных видов работ, связанных с будущей профессиональной деятельностью. Производственная практика проходила в Московском университете им. С.Ю. Витте (далее - Университете) на кафедре информационных систем.

# **Цель практики**

приобретение первичных профессиональных навыков, практического опыта, закрепление, систематизация и расширение теоретических знаний по дисциплинам учебного плана при прохождении учебной практики в Университете.

# **Задачи практики**

1. Укрепление полученных теоретических знаний в анализе данных и программировании на Python через простой анализ данных с вычислением основных числовых характеристик и их визуализацией.

2. Освоение навыков научного поиска и практической работы с информационными источниками данных путем анализа данных о продажах на онлайн-площадках.

3. Развитие практических методик постановки целей и задач для групповых проектов и управления процессом изучения.

4. Получение навыков работы в команде и участия в групповых проектах, включая распределение ролей и взаимодействие в команде при анализе данных о продажах на площадках.

5. Усовершенствование навыков использования иностранного языка при анализе информации из зарубежных интернет-ресурсов о продажах на маркетплейсах.

6. Овладение применением современных информационных технологий и программных инструментов, включая библиотеку NumPy, для решения задач анализа данных о продажах на площадках.

7. Овладение навыками установки программного обеспечения, включая необходимые библиотеки и инструменты для работы с данными о продажах на площадках.

8. Формирование практических навыков алгоритмизации вычислительных процессов и программирования для обработки данных о продажах, их отладки и тестирования на языке программирования Python.

9. Приобретение навыков экспериментального исследования больших объемов данных и практического решения информационных задач по обработке данных о продажах.

10. Получение навыков выполнения инженерных расчетов, связанных с анализом данных о продажах на площадках.

11. Развитие самостоятельной работы и самоорганизации в выполнении задач анализа данных о продажах на площадках с использованием Python и NumPy.

12. Формирование навыков составления отчетной документации по результатам практической работы.

Источниками информации для выполнения практики служили открытые информационные базы, информационно-справочные документы, интернет-ресурсы и учебно-методические материалы, включая информацию, представленную на официальном сайте университета в разделе «Сведения об образовательной организации»: https://www.muiv.ru/sveden/.

# **Выбор платформы и библиотек**

# **Обоснование выбора платформы интерпретатора языка Python**

1. Python - язык программирования:

- Python является одним из наиболее популярных и широко используемых языков программирования в области анализа данных и машинного обучения.

- Python обладает простым и понятным синтаксисом, что делает его доступным для новичков в программировании и упрощает разработку и понимание кода.

- Python обладает обширной экосистемой библиотек и инструментов для анализа данных, визуализации и научных вычислений, что делает его идеальным выбором для работы с данными о продажах и их анализа.

2. Среда разработки PyCharm:

- PyCharm является мощной и удобной средой разработки, специально разработанной для работы с Python.

- PyCharm предоставляет широкий набор инструментов, редактор кода с подсветкой синтаксиса, автодополнением и проверкой ошибок, а также отладчик, который значительно упрощает процесс разработки и отладки программного кода на Python.

- PyCharm обладает интегрированной поддержкой виртуальных окружений Python, что позволяет создавать изолированные среды для работы с разными версиями Python и библиотеками, обеспечивая стабильность и совместимость проекта.

3. Библиотеки Matplotlib, Pandas и NumPy:

- Matplotlib является одной из основных библиотек для визуализации данных в Python. Она предоставляет множество инструментов для создания различных графиков, диаграмм и визуальных представлений, что позволяет наглядно представить результаты анализа данных о продажах.

- Pandas предоставляет высокоуровневые структуры данных и инструменты для манипулирования и анализа данных. Она упрощает загрузку, обработку, фильтрацию и агрегацию данных о продажах с маркетплейсов, а также предоставляет мощные средства для работы с временными рядами и таблицами данных.

- NumPy является фундаментальной библиотекой для работы с числовыми данными в Python. Она предоставляет эффективные структуры данных, такие как многомерные массивы, и мощные математические функции для выполнения вычислений с данными о продажах, такие как вычисление статистических показателей и числовых характеристик.

Сочетание языка Python, среды разработки PyCharm и библиотек Matplotlib, Pandas и NumPy обеспечивает эффективные инструменты для анализа данных о продажах с маркетплейсов, обработки информации, визуализации результатов и проведения статистических вычислений, что делает их оптимальным выбором для данной работы.

**Выбор необходимых библиотек**

Сегодня в мире программирования на языке Python существует множество библиотек, предназначенных для работы с данными. Однако, лишь некоторые из них сочетают в себе удобство использования, богатый функционал и поддержку со стороны разработчиков. В данной программе были выбраны следующие библиотеки:

1. Pandas (pd):

- Pandas предоставляет удобные и эффективные структуры данных, такие как DataFrame и Series, для работы с данными.

- Она предоставляет мощные функции для чтения, обработки, фильтрации и агрегации данных.

- В данной программе библиотека Pandas используется для работы с данными о продажах с маркетплейсов, выполнения статистических вычислений и создания результирующих таблиц.

2. Matplotlib.pyplot (plt):

- Matplotlib является одной из основных библиотек для визуализации данных в Python.

- Она предоставляет множество функций для создания различных типов графиков и диаграмм.

- В данной программе библиотека Matplotlib используется для построения графиков, таких как линейные графики и гистограммы.

3. Numpy (np):

- Numpy является фундаментальной библиотекой для работы с числовыми данными в Python.

- Она предоставляет эффективные структуры данных, такие как многомерные массивы, и мощные функции для выполнения математических операций.

- В данной программе библиотека Numpy используется для генерации случайных чисел, работы с матрицами и векторами, а также выполнения вычислений.

4. Random:

- Модуль random является частью стандартной библиотеки Python.

- Он предоставляет функции для генерации псевдослучайных чисел.

- В данной программе модуль random используется для генерации случайных чисел в списке.

5. Os:

- Модуль os также является частью стандартной библиотеки Python.

- Он предоставляет функции для работы с операционной системой, такие как чтение файлов и получение информации о директориях.

- В данной программе модуль os используется для загрузки датасетов и получения списка файлов.

Выбор данных библиотек обусловлен их популярностью, удобством использования и мощными функциональными возможностями. Они предоставляют инструменты, необходимые для чтения, обработки, анализа данных и визуализации результатов.

# **Вывод по разделу**

Выбор языка программирования Python, среды разработки PyCharm и использование библиотек pandas, matplotlib и numpy оказались успешными для анализа данных о продажах с маркетплейсов. Эти инструменты обеспечили удобство, эффективность и функциональность в работе с данными, а также возможность визуализации результатов анализа.

Основываясь на информации в данном разделе, можно сделать следующие выводы (табл.2.1).

Таблица 2.1

Выводы по разделу 2

|  |  |  |
| --- | --- | --- |
| Выводы | Код и наименование компетенции выпускника программы бакалавриата | Код и наименование индикатора достижения компетенции |
| После проведения анализа сред разработки и необходимой информационно-технологической инфраструктуры, мы сделали следующие выводы:  В соответствии с индивидуальным заданием, была выбрана современная платформа программирования Python, которая предоставляет множество возможностей для решения прикладных профессиональных задач. Python является одним из наиболее популярных языков программирования в области анализа данных, благодаря своей простоте и гибкости.  Для выполнения поставленных задач были выбраны библиотеки Pandas и Matplotlib. Библиотека Pandas предоставляет высокоуровневые структуры данных, такие как DataFrame, которые облегчают обработку и анализ больших объемов данных. Библиотека Matplotlib, в свою очередь, позволяет создавать графики и визуализировать данные, что помогает наглядно представить результаты анализа.  При анализе технической документации к среде разработки и используемым библиотекам было обращено внимание на доступность информации как на русском, так и на иностранных языках. Это позволило получить полное понимание особенностей и функциональности данных инструментов.  Выбор данных платформы и библиотек был основан на их популярности, широкой поддержке сообщества разработчиков, а также на их возможностях в области анализа данных. В результате использования данных инструментов была достигнута эффективность, точность и удобство работы с данными о продажах с маркетплейсов. | ОПК-1. Способен проводить моделирование, анализ и совершенствование бизнес-процессов и информационно-технологической инфраструктуры предприятия в интересах достижения его стратегических целей с использованием современных методов и программного инструментария; | ОПК-1.1.  Знать методы анализа бизнес-процессов предприятия  ОПК-1.2.  Уметь использовать программный инструментарий для моделирования бизнес-процессов  ОПК-1.3.  Владеть навыками анализа и модернизации информационно-технологической инфраструктуры предприятия. |
| ОПК-2. Способен проводить исследование и анализ рынка информационных систем и информационно-коммуникационных технологий, выбирать рациональные решения для управления бизнесом; | ОПК-2.1.  Знать методы исследования качества, сложности и надежности информационных систем  ОПК-2.2.  Уметь проводить оценку эффективности внедрения информационных систем и информационно-коммуникационных технологий управления бизнесом.  ОПК-2.3.  Владеть навыками анализа рынка информационных систем и информационно-коммуникативных технологий. |
| УК-4. Способен осуществлять деловую коммуникацию в устной и письменной формах на государственном языке Российской Федерации и иностранном(ых) языке(ах) | УК-4.1.  Знает принципы построения устного и письменного высказывания на государственном и иностранном языках; требования к деловой устной и письменной коммуникации.  УК-4.2.  Умеет вести академическую и профессиональную дискуссии на государственном языке Российской Федерации и иностранном(ых) языке(ах).  УК-4.3.  Владеет методикой составления суждения в межличностном деловом общении на государственном и иностранном языках. |
| Были произведены установка и подключение необходимых библиотек в среду разработки Python с целью использования их в качестве готовых компонентов при разработке алгоритмов и программ. Это включало установку библиотек pandas, matplotlib и numpy, которые предоставляют мощные функциональные возможности для обработки и анализа данных, визуализации результатов и выполнения математических операций. | ОПК-3. Способен управлять процессами создания и использования продуктов и услуг в сфере информационно-коммуникационных технологий, в том числе разрабатывать алгоритмы и программы для их практической реализации; | ОПК-3.1.  Знать методы проектирования и развертывания программных продуктов и услуг, обеспечивающих поддержку бизнес-процессов предприятия.  ОПК-3.2.  Уметь использовать современные технологические платформы и средства разработки для создания программных продуктов и услуг в сфере информационно-коммуникационных технологий.  ОПК-3.3.  Владеть методами проектирования и развертывания продуктов и услуг в сфере информационно-коммуникационных технологий. |

# **Простейший анализ данных**

# **Получение данных. Создание списка со случайных чисел**

Ниже приведен пример кода, который демонстрирует этот процесс:

|  |
| --- |
| import random  # Генерация случайных чисел и создание списка  numbers = [random.randint(-10000, 10000) for \_ in range(1000)]  print(numbers) |

Объяснение кода:

1. Мы импортируем модуль `random`, который предоставляет функционал для работы со случайными числами.

2. Мы используем генератор списка и цикл `for` для создания списка `numbers` из 1000 случайных чисел.

3. Функция `random.randint(a, b)` генерирует случайное целое число в диапазоне от `a` до `b`, включая оба конца.

4. Результат списка `numbers` выводится на экран с помощью функции `print()`.

Таким образом, код генерирует список `numbers` со случайными числами в диапазоне от -10000 до 10000. Этот список может быть использован для дальнейшего анализа данных, расчета статистических характеристик и визуализации.

# **Определение элементарных числовых характеристик**

Ниже приведен пример кода, который демонстрирует этот процесс:

|  |
| --- |
| import random  import numpy as np  import pandas as pd  # Генерация случайных чисел и создание списка  numbers = [random.randint(-10000, 10000) for \_ in range(1000)]  # Преобразование списка в DataFrame для удобной работы с данными  df = pd.DataFrame(numbers, columns=['Numbers'])  # Вычисление элементарных числовых характеристик  min\_value = df['Numbers'].min()  max\_value = df['Numbers'].max()  mean\_value = df['Numbers'].mean()  std\_value = df['Numbers'].std()  sum\_value = df['Numbers'].sum()  # Вывод результатов  print("Минимальное значение:", min\_value)  print("Максимальное значение:", max\_value)  print("Среднее значение:", mean\_value)  print("Стандартное отклонение:", std\_value)  print("Сумма значений:", sum\_value) |

Объяснение кода:

1. Мы импортируем модули `random`, `numpy` и `pandas`, необходимые для генерации случайных чисел, выполнения математических операций и работы с данными в табличном формате.

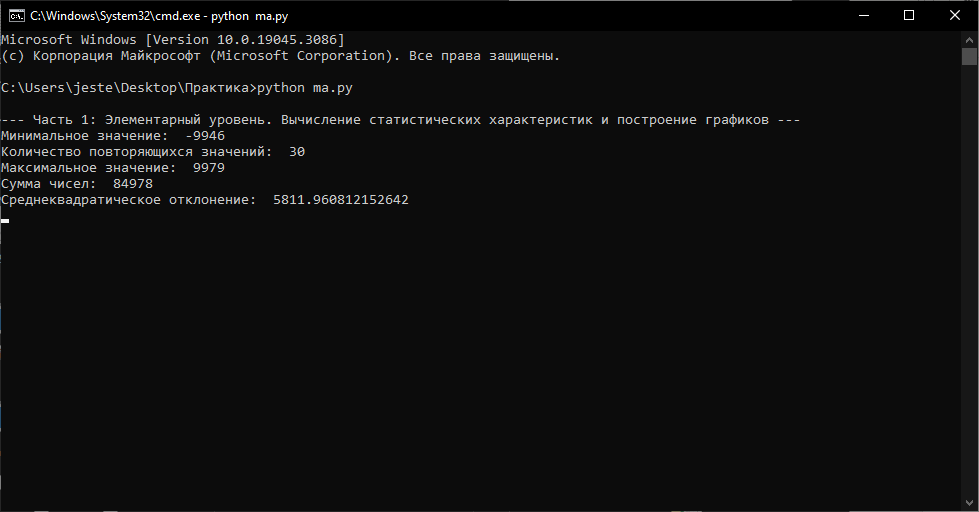
2. Мы используем генератор списка и цикл `for` для создания списка `numbers` из 1000 случайных чисел.

3. Мы преобразуем список `numbers` в объект DataFrame с помощью функции `pd.DataFrame()`, чтобы удобно работать с данными.

4. Мы используем функции DataFrame для вычисления элементарных числовых характеристик: минимального значения (`min()`), максимального значения (`max()`), среднего значения (`mean()`), стандартного отклонения (`std()`) и суммы значений (`sum()`).

5. Результаты выводятся на экран с помощью функции `print()`.

Пример работы программы:



Таким образом, код позволяет определить элементарные числовые характеристики для списка случайных чисел, включая минимальное и максимальное значения, среднее значение, стандартное отклонение и сумму значений. Это полезно для анализа данных и получения основных статистических показателей.

# **Визуализирование данных с помощью стандартных библиотек**

Ниже приведен пример кода, который демонстрирует этот процесс:

|  |
| --- |
| import random  import matplotlib.pyplot as plt  # Генерация случайных чисел и создание списка  numbers = [random.randint(-10000, 10000) for \_ in range(1000)]  # Линейный график  plt.figure(figsize=(10, 6)) # Устанавливаем размер графика  plt.plot(df['Numbers']) # Строим линейный график  plt.title('Линейный график') # Заголовок графика  plt.show() # Отображение графика  # Гистограмма  plt.figure(figsize=(10, 6)) # Устанавливаем размер графика  plt.hist(df\_rounded, bins=30) # Строим гистограмму с 30 бинами  plt.title('Гистограмма') # Заголовок графика  plt.show() # Отображение графика  # Линейный график отсортированных значений по возрастанию  df\_sorted\_asc = df.sort\_values(by='Numbers') # Сортируем данные по возрастанию  plt.figure(figsize=(10, 6)) # Устанавливаем размер графика  plt.plot(df\_sorted\_asc['Numbers'].reset\_index(drop=True)) # Строим линейный график  plt.title('Линейный график отсортированных значений по возрастанию') # Заголовок графика  plt.show() # Отображение графика  # Линейный график отсортированных значений по убыванию  df\_sorted\_desc = df.sort\_values(by='Numbers', ascending=False) # Сортируем данные по убыванию  plt.figure(figsize=(10, 6)) # Устанавливаем размер графика  plt.plot(df\_sorted\_desc['Numbers'].reset\_index(drop=True)) # Строим линейный график  plt.title('Линейный график отсортированных значений по убыванию') # Заголовок графика  plt.show() # Отображение графика |

Каждый график выполняет свою уникальную функцию:

1. Линейный график - отображает значения списка чисел на оси y в порядке их следования по оси x.

2. Гистограмма - показывает распределение частоты встречаемости значений из списка чисел в виде столбцовых бинов.

3. Линейный график отсортированных значений по возрастанию - представляет собой линейный график, где значения списка чисел отсортированы по возрастанию.

4. Линейный график отсортированных значений по убыванию - аналогично предыдущему графику, но значения отсортированы по убыванию.

Каждый график позволяет визуализировать данные с разными уровнями детализации и подчеркнуть различные аспекты распределения или порядка значений в списке чисел.

Объяснение кода:

1. Мы импортируем модули `random` и `matplotlib.pyplot`, необходимые для генерации случайных чисел и визуализации данных.

2. Мы используем генератор списка и цикл `for` для создания списка `numbers` из 1000 случайных чисел.

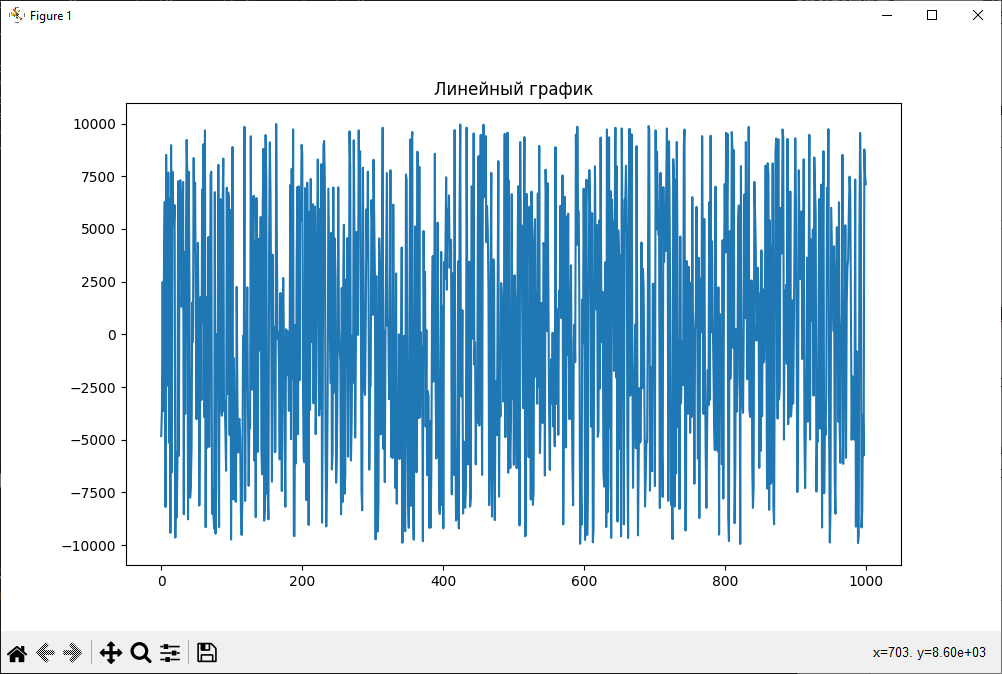
3. Мы создаем линейный график с помощью функции `plt.plot()`, передавая список `numbers` в качестве данных.

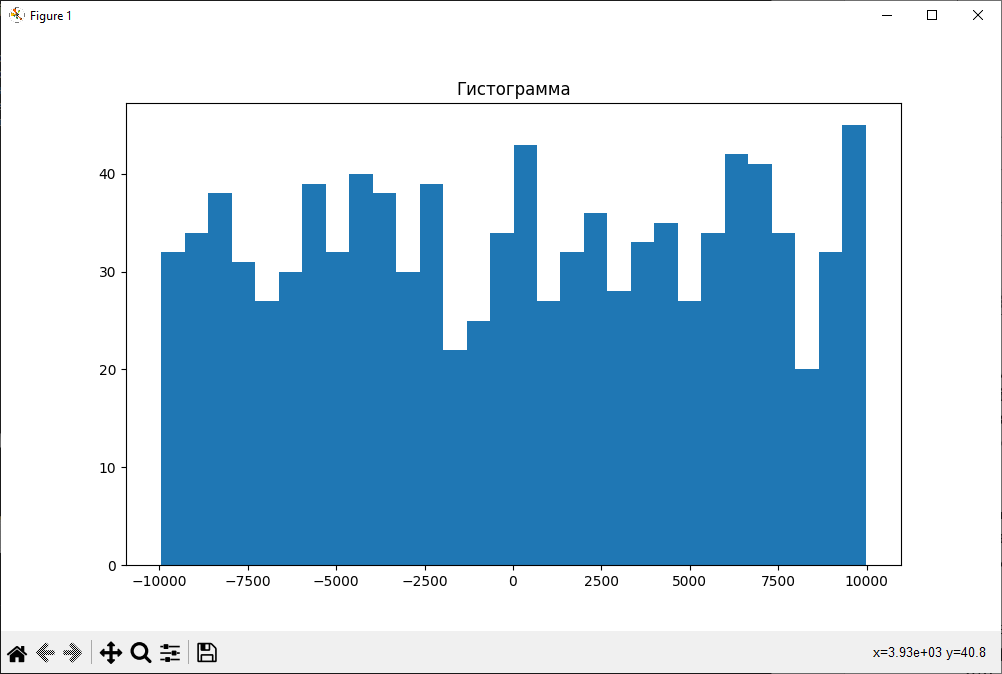
4. Мы добавляем заголовок графика с помощью функции `plt.title()`.

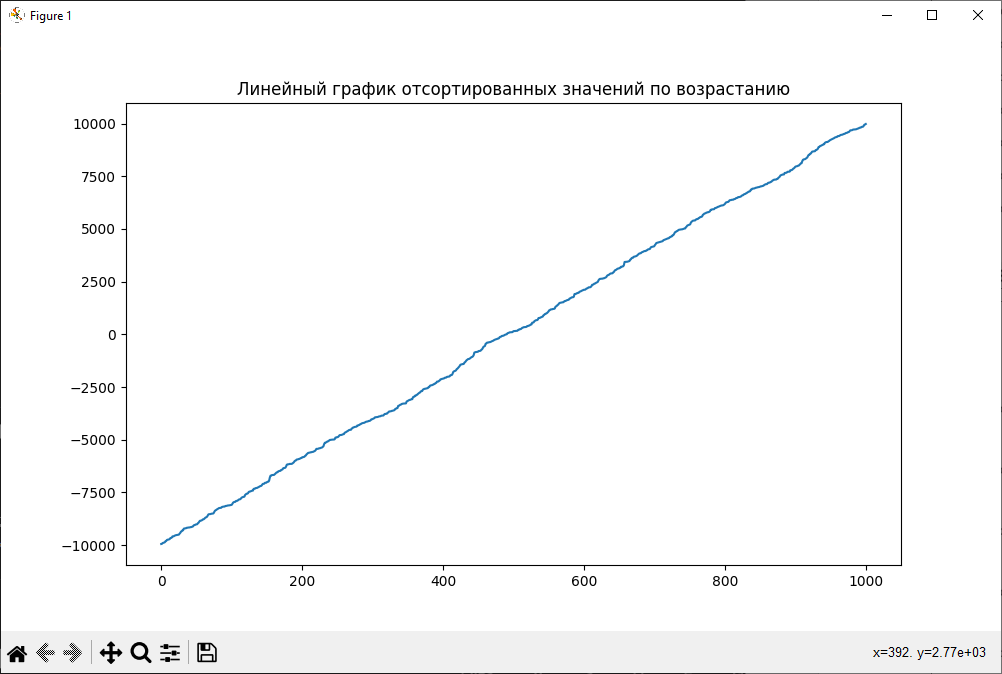
5. Мы добавляем метки осей x и y с помощью функций `plt.xlabel()` и `plt.ylabel()`.

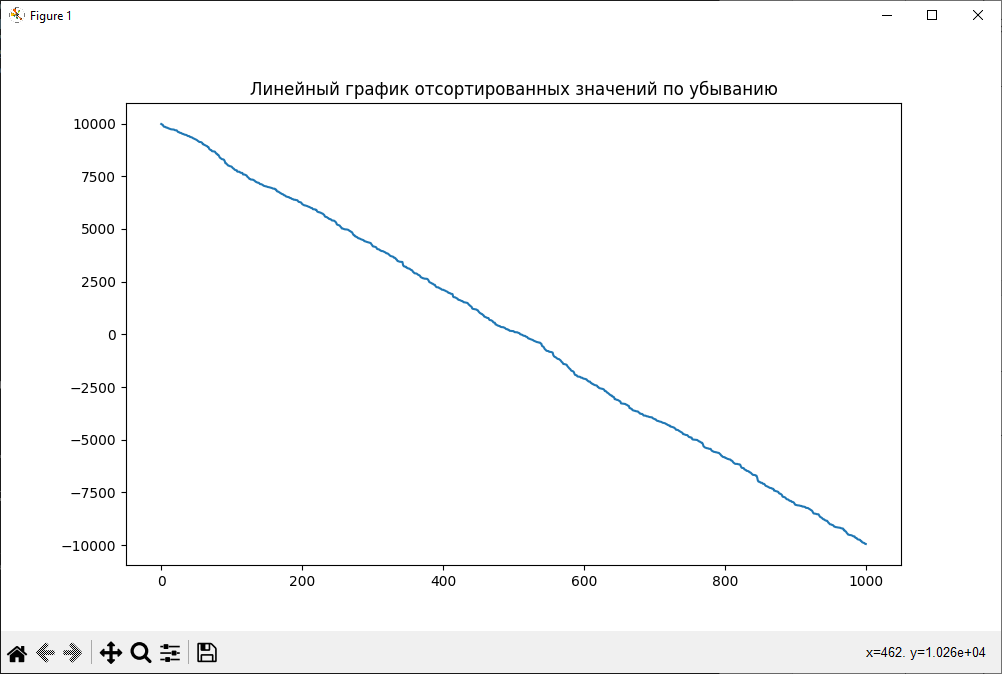
6. Мы отображаем график на экране с помощью функции `plt.show()`.

Пример работы программы:









Таким образом, представленный код позволяет визуализировать данные в виде линейного графика. Этот подход может быть расширен для создания других типов графиков, включая гистограммы, диаграммы рассеяния и многое другое, с использованием функций и возможностей библиотеки `matplotlib`.

# **Выводы по разделу**

В данном разделе были рассмотрены основные этапы простейшего анализа данных с использованием языка программирования Python.

Первым этапом является получение данных. В примере было показано, как сгенерировать список случайных чисел с использованием модуля `random`. Этот шаг демонстрирует возможность получения данных и создания набора числовых значений для дальнейшего анализа.

Затем был проведен анализ элементарных числовых характеристик этого списка чисел. С использованием библиотеки `pandas` были вычислены основные статистические характеристики, такие как минимальное и максимальное значения, количество дубликатов, сумма и среднеквадратическое отклонение. Это позволяет получить общее представление о данных и их основных свойствах.

Для визуализации данных были использованы стандартные библиотеки `matplotlib.pyplot`. Примеры включают построение линейного графика, гистограммы и линейных графиков отсортированных значений по возрастанию и убыванию. Визуализация данных позволяет лучше понять и представить распределение значений, их тренды и взаимосвязи.

Таким образом, простейший анализ данных включает получение данных, создание списка со случайными числами, определение элементарных числовых характеристик и визуализацию данных с использованием стандартных библиотек. Эти шаги помогают в изучении данных, обнаружении основных закономерностей и представлении результатов анализа в понятной и наглядной форме.

Основываясь на информации в данном разделе можно сделать следующие выводы (табл.2.1).

Таблица 2.1

Выводы по разделу 2

|  |  |  |
| --- | --- | --- |
| Выводы | Код и наименование компетенции выпускника программы бакалавриата | Код и наименование индикатора достижения компетенции |
| В ходе работы был осуществлен поиск внешних ресурсов с данными в соответствии с индивидуальным заданием. Были выбраны подходящие источники, содержащие необходимую информацию. Предварительный анализ данных из этих внешних источников позволил сформировать наборы случайных данных, которые были использованы в дальнейшей работе.  На основе системного подхода и анализа структуры этих внешних данных, был выбран подходящий формат хранения данных. Это позволило эффективно организовать и структурировать данные для последующего анализа и обработки. Выбранный формат обеспечивает удобство доступа к данным и упрощает выполнение операций с ними.  В результате проведенных действий были получены данные, готовые для дальнейшего использования в анализе и обработке с использованием выбранных инструментов и методов. | УК-1. Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач | УК-1.1.  Знает принципы сбора, отбора и обобщения информации, методики системного подхода для решения профессиональных задач.  УК-1.2.  Умеет анализировать и систематизировать разнородные данные, оценивать эффективность процедур анализа проблем и принятия решений в профессиональной деятельности.  УК-1.3.  Владеет навыками научного поиска и практической работы с информационными источниками; методами принятия решений. |
| В рамках данного раздела была выполнена постановка задачи и сформулирована цель исследования данных, что позволило четко определить направление работы. Также была сформирована проектная группа, в которой каждому участнику были распределены функциональные обязанности, определен объем работ и заданы временные ограничения на разработку.  Выбор оптимальных способов решения поставленных задач был осуществлен с учетом имеющихся возможностей каждого разработчика. Это позволило эффективно использовать ресурсы и компетенции команды, обеспечивая более эффективную разработку программного кода и достижение поставленных целей.  Таким образом, в результате выполнения данного раздела была сформирована ясная постановка задачи, определена цель исследования данных, распределены функциональные обязанности и выбраны оптимальные способы решения задач. Это создает основу для успешного выполнения проекта и достижения желаемых результатов. | УК-2. Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений | УК-2.1.  Знает необходимые для осуществления профессиональной деятельности правовые нормы и методологические основы принятия управленческого решения.  УК-2.2.  Умеет анализировать альтернативные варианты решений для достижения намеченных результатов; разрабатывать план, определять целевые этапы и основные направления работ.  УК-2.3.  Владеет методиками разработки цели и задач проекта; методами оценки продолжительности и стоимости проекта, а также потребности в ресурсах. |
| УК-3. Способен осуществлять социальное взаимодействие и реализовывать свою роль в команде | УК-3.1.  Знает типологию и факторы формирования команд, способы социального взаимодействия.  УК-3.2.  Умеет действовать в духе сотрудничества; принимать решения с соблюдением этических принципов их реализации; проявлять уважение к мнению и культуре других; определять цели и работать в направлении личностного, образовательного и профессионального роста.  УК-3.3.  Владеет навыками распределения ролей в условиях командного взаимодействия; методами оценки своих действий, планирования и управления временем. |
| УК-6. Способен управлять своим временем, выстраивать и реализовывать траекторию саморазвития на основе принципов образования в течение всей жизни | УК-6.1.  Знает основные принципы самовоспитания и самообразования, исходя из требований рынка труда.  УК-6.2.  Умеет демонстрировать умение самоконтроля и рефлексии, позволяющие самостоятельно корректировать обучение по выбранной траектории.  УК-6.3.  Владеет способами управления своей познавательной деятельностью и удовлетворения образовательных интересов и потребностей. |
| УК-5. Способен воспринимать межкультурное разнообразие общества в социально-историческом, этическом и философском контекстах | УК-5.1.  Знает основные категории философии, законы исторического развития, основы межкультурной коммуникации.  УК-5.2.  Умеет вести коммуникацию в мире культурного многообразия и  демонстрировать взаимопонимание между обучающимися – представителями различных культур с соблюдением этических и межкультурных норм.  УК-5.3.  Владеет практическими навыками анализа философских и исторических фактов, оценки явлений культуры; способами анализа и пересмотра своих взглядов в случае разногласий и конфликтов в межкультурной коммуникации. |
|
|
| - Определены объемы работы по каждой задаче.  - Заданы сроки представления результатов создания своего кода.  - Установлены временные траектории совместной работы над общими фрагментами кода с учетом требований безопасности и режима труда и отдыха.  В результате выполнения данного раздела были определены объемы работы, заданы сроки представления результатов и установлены временные траектории совместной работы. Это обеспечивает организацию и эффективность работы команды в рамках проекта. | УК-6. Способен управлять своим временем, выстраивать и реализовывать траекторию саморазвития на основе принципов образования в течение всей жизни | УК-6.1.  Знает основные принципы самовоспитания и самообразования, исходя из требований рынка труда.  УК-6.2.  Умеет демонстрировать умение самоконтроля и рефлексии, позволяющие самостоятельно корректировать обучение по выбранной траектории.  УК-6.3.  Владеет способами управления своей познавательной деятельностью и удовлетворения образовательных интересов и потребностей. |
| УК-7. Способен поддерживать должный уровень физической подготовленности для обеспечения полноценной социальной и профессиональной деятельности | УК-7.1.  Знает виды физических упражнений; научно-практические основы физической культуры и здорового образа и стиля жизни.  УК-7.2.  Умеет применять на практике разнообразные средства физической культуры, спорта и туризма для сохранения и укрепления здоровья, психофизической подготовки и самоподготовки к будущей жизни и профессиональной деятельности; использовать творчески средства и методы физического воспитания для профессионально-личностного развития, физического самосовершенствования, формирования здорового образа и стиля жизни.  УК-7.3.  Владеет средствами и методами укрепления индивидуального здоровья, физического самосовершенствования. |
| УК-8. Способен создавать и поддерживать в повседневной жизни и в профессиональной деятельности безопасные условия жизнедеятельности для сохранения природной среды, обеспечения устойчивого развития, в том числе при угрозе и возникновении чрезвычайных ситуаций и военных конфликтов | УК-8.1.  Знает теоретические основы жизнедеятельности в системе «человек – среда обитания», методы защиты в условиях чрезвычайных ситуаций, военных конфликтов;  методы сохранения природной среды, факторы обеспечения устойчивого развития общества.  УК-8.2.  Умеет идентифицировать опасные и вредные факторы в рамках осуществляемой деятельности; оценивать возможные риски появления опасностей и чрезвычайных ситуаций; создавать и поддерживать безопасные условия жизнедеятельности в повседневной жизни и в профессиональной деятельности.  УК-8.3.  Владеет методами прогнозирования возникновения опасных или чрезвычайных ситуаций; навыками по применению основных методов защиты в условиях чрезвычайных ситуаций в повседневной жизни и профессиональной деятельности. |
| УК-9. Способен использовать базовые дефектологические знания в социальной и профессиональной сферах | УК-9.1 Знает понятия инклюзивного взаимодействия при коммуникации в различных сферах жизнедеятельности, с учетом социально-психологических особенностей лиц, имеющими ограниченные возможности здоровья или инвалидность  УК-9.2. Умеет вести профессиональную деятельность с лицами имеющими ограниченные возможности здоровья или инвалидность  УК-9.3 Владеет навыками взаимодействия с лицами имеющими ограниченные возможности здоровья или инвалидность в социальной и профессиональной сферах |
| УК-10. Способен принимать обоснованные экономические решения в различных областях жизнедеятельности | УК-10.1 Знает основные документы, регламентирующие экономическую деятельность; источники финансирования профессиональной деятельности; принципы планирования экономической деятельности  УК-10.2 Умеет обосновывать принятие экономических  решений, использовать методы экономического планирования для достижения поставленных целей  УК-10.3 Владеет навыками применения экономических инструментов |
| УК-11. Способен формировать нетерпимое отношение к коррупционному поведению | УК-11.1 Знает действующие правовые нормы, обеспечивающие борьбу с коррупцией в различных областях жизнедеятельности; способы профилактики коррупции и формирования нетерпимого отношения к ней  УК-11.2 Умеет планировать, организовывать и проводить мероприятия, обеспечивающие формирование гражданской позиции и предотвращение коррупции в обществе  УК-11.3 Владеет навыками взаимодействия в обществе на основе нетерпимого отношения к коррупции |

# **Работа с библиотекой Numpy**

# **Генерация массивов данных**

Ниже приведен код с комментариями, описывающими этот процесс:

|  |
| --- |
| # Генерация квадратного массива с помощью функции np.random.randint  array = np.random.randint(low=-100, high=100, size=(sum\_id, sum\_id)) |

Данная часть кода использует функцию `np.random.randint` из библиотеки `numpy` для генерации квадратного массива данных. Параметры функции задаются следующим образом:

- `low` - минимальное значение элементов массива

- `high` - максимальное значение элементов массива

- `size` - размеры массива, указывающие количество строк и столбцов

В данном случае, генерируется квадратный массив размером `sum\_id x sum\_id`, где `sum\_id` - это сумма цифр исходного значения `id`. Значения элементов массива выбираются случайным образом в заданном диапазоне от `low` до `high`.

Генерация квадратного массива позволяет создать искусственные данные для выполнения различных математических операций, таких как вычисление определителя матрицы, решение матричных уравнений и других операций, основанных на массивах данных.

# **Нахождение определителя А**

Ниже приведен код с комментариями, описывающими этот процесс:

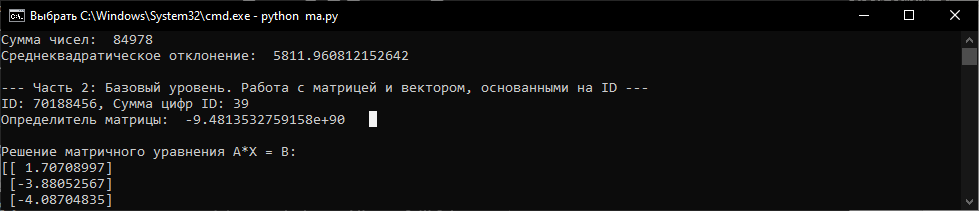
|  |
| --- |
| # Вычисление определителя матрицы, используя функцию np.linalg.det  try:  det = np.linalg.det(array)  print("Определитель матрицы: ", det)  except np.linalg.LinAlgError:  print("Определитель не может быть вычислен") |

Для вычисления определителя матрицы A используется функция `np.linalg.det` из библиотеки `numpy`. Данная функция принимает матрицу в качестве аргумента и возвращает ее определитель.

В представленном коде, определитель матрицы вычисляется с помощью функции `np.linalg.det(array)`, где `array` представляет собой квадратную матрицу, полученную ранее. Вычисленное значение определителя сохраняется в переменной `det`.

В блоке `try-except` осуществляется обработка исключения `np.linalg.LinAlgError`, которое может возникнуть, если матрица `array` не обладает обратимостью или другими свойствами, необходимыми для вычисления определителя. В случае возникновения исключения, выводится сообщение "Определитель не может быть вычислен".

Пример работы программы:



Вычисление определителя матрицы является важной операцией в линейной алгебре и позволяет оценить свойства и характеристики матрицы, такие как ее обратимость и линейная независимость векторов.

# **Генерация Вектор-Столбца**

Ниже приведен код с комментариями, описывающими этот процесс:

|  |
| --- |
| # Генерация вектор-столбца размером sum\_id  B = np.random.randint(low=-100, high=100, size=(sum\_id, 1)) |

Данная часть кода использует функцию `np.random.randint` из библиотеки `numpy` для генерации вектор-столбца B. Параметры функции задаются следующим образом:

- `low` - минимальное значение элементов вектора

- `high` - максимальное значение элементов вектора

- `size` - размеры вектора, указывающие количество элементов и его ориентацию (в данном случае `(sum\_id, 1)` означает, что вектор является столбцом)

В данном случае, генерируется вектор-столбец B размером `sum\_id`, где `sum\_id` - это сумма цифр исходного значения `id`. Значения элементов вектора выбираются случайным образом в заданном диапазоне от `low` до `high`.

Генерация вектор-столбца позволяет создать искусственные данные для выполнения различных операций, таких как решение матричных уравнений, умножение матрицы на вектор и других операций, связанных с линейной алгеброй.

# **Решение матричного уравнения**

Ниже приведен код с комментариями, описывающими этот процесс:

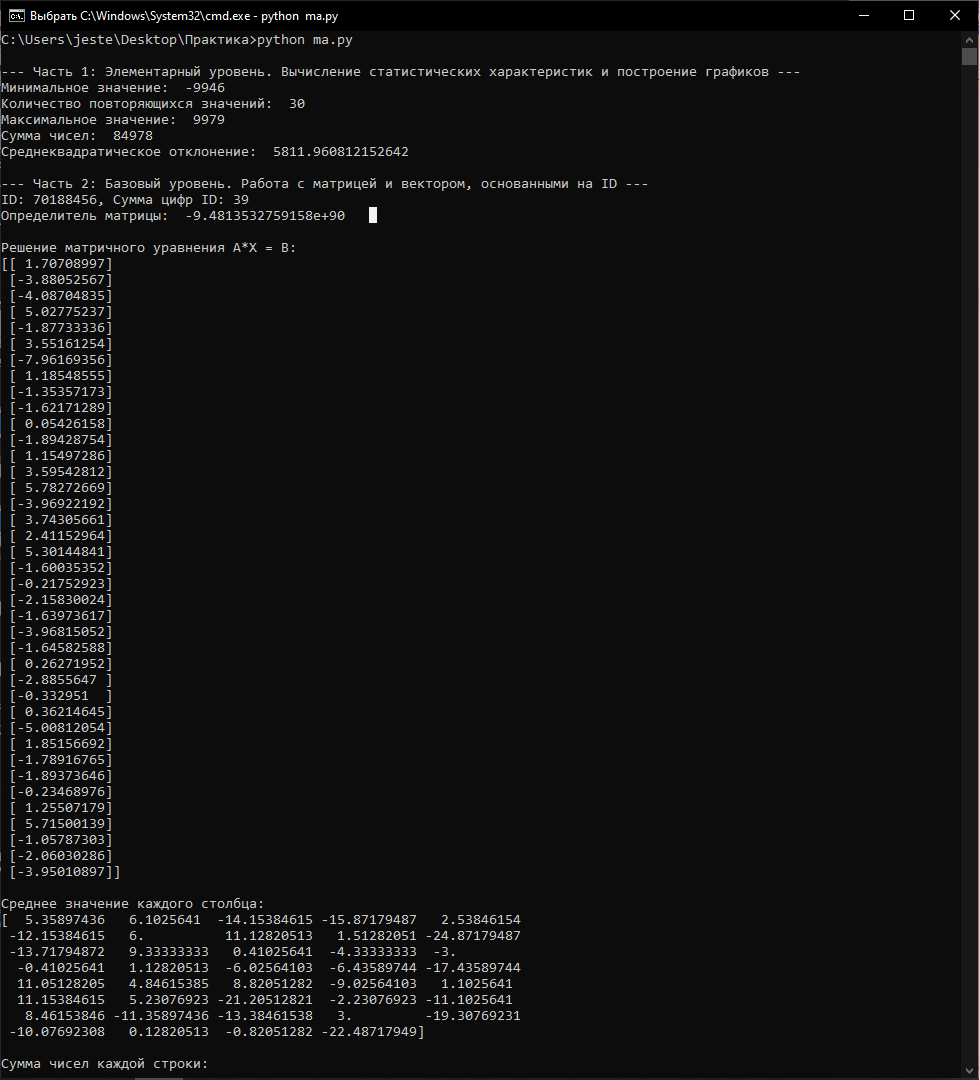
|  |
| --- |
| # Решение матричного уравнения A \* X = B, используя функцию np.linalg.solve  try:  X = np.linalg.solve(array, B)  print("\nРешение матричного уравнения A \* X = B:")  print(X)  except np.linalg.LinAlgError:  print("Матричное уравнение не может быть решено") |

Для решения матричного уравнения A \* X = B используется функция `np.linalg.solve` из библиотеки `numpy`. Данная функция принимает два аргумента: матрицу A и вектор-столбец B, и возвращает решение матричного уравнения X.

В представленном коде, решение матричного уравнения вычисляется с помощью функции `np.linalg.solve(array, B)`, где `array` представляет собой квадратную матрицу, полученную ранее, а `B` - вектор-столбец. Результат решения сохраняется в переменной `X`.

В блоке `try-except` осуществляется обработка исключения `np.linalg.LinAlgError`, которое может возникнуть, если матрица `array` не обладает свойствами, необходимыми для решения матричного уравнения (например, матрица является вырожденной). В случае возникновения исключения, выводится сообщение "Матричное уравнение не может быть решено".

Пример работы программы:



Решение матричного уравнения позволяет найти значения неизвестных (элементов вектора-столбца X), удовлетворяющих заданному матричному соотношению. Это имеет важное значение в линейной алгебре и находит применение во многих областях, таких как обработка сигналов, машинное обучение, физика и других науках.

# **Рассчёт средней арифметической**

Ниже приведен код с комментариями, описывающими этот процесс:

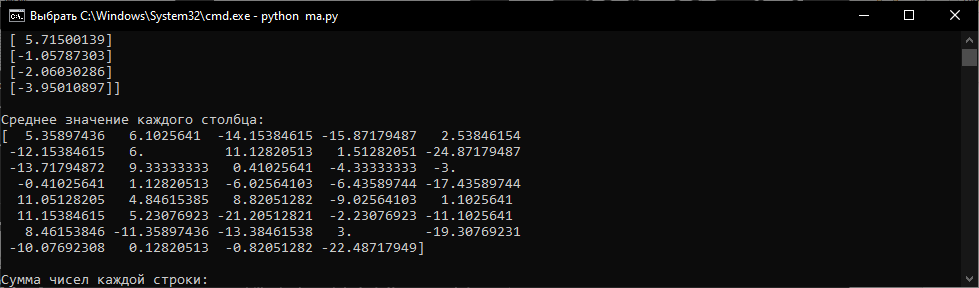
|  |
| --- |
| # Вычисление среднего значения каждого столбца и суммы каждой строки, используя функции np.mean и np.sum  mean\_values = np.mean(array, axis=0)  print("\nСреднее значение каждого столбца:")  print(mean\_values)  sum\_values = np.sum(array, axis=1)  print("\nСумма чисел каждой строки:")  print(sum\_values) |

Для вычисления среднего значения каждого столбца матрицы и суммы каждой строки используются функции `np.mean` и `np.sum` из библиотеки `numpy`. Обе функции принимают два аргумента: матрицу и ось, по которой должны быть вычислены значения.

В представленном коде, среднее значение каждого столбца вычисляется с помощью функции `np.mean(array, axis=0)`, где `array` - это матрица, полученная ранее, а `axis=0` указывает на ось столбцов. Результат вычислений сохраняется в переменной `mean\_values`.

Аналогично, сумма чисел каждой строки вычисляется с помощью функции `np.sum(array, axis=1)`, где `axis=1` указывает на ось строк. Результат вычислений сохраняется в переменной `sum\_values`.

Пример работы программы:



Расчет среднего значения каждого столбца и суммы каждой строки является важной числовой характеристикой данных. Эти значения могут дать представление о распределении данных в матрице и помочь в анализе и интерпретации результатов.

# **Рассчёт суммы числе по каждой строке**

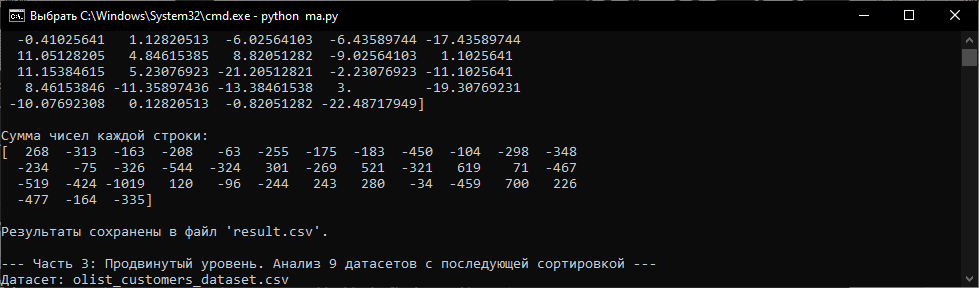
Ниже приведен код с комментариями, описывающими этот процесс:

|  |
| --- |
| # Вычисление суммы чисел каждой строки, используя функцию np.sum  sum\_values = np.sum(array, axis=1)  print("\nСумма чисел каждой строки:")  print(sum\_values) |

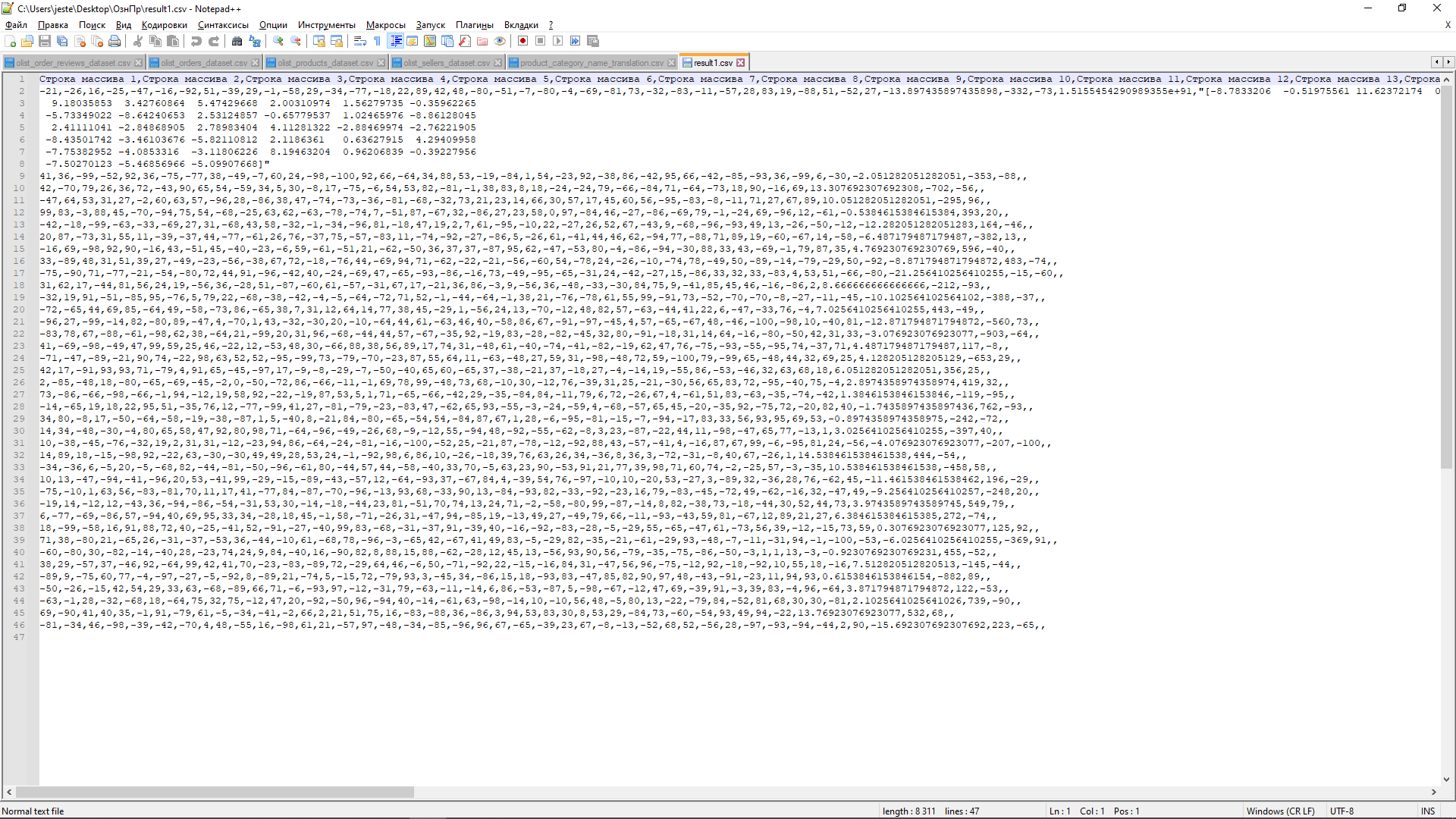
Для вычисления суммы чисел каждой строки в матрице используется функция `np.sum` из библиотеки `numpy`. Функция принимает два аргумента: матрицу и ось, по которой должна быть вычислена сумма.

В представленном коде, сумма чисел каждой строки вычисляется с помощью функции `np.sum(array, axis=1)`, где `array` - это матрица, полученная ранее, а `axis=1` указывает на ось строк. Результат вычислений сохраняется в переменной `sum\_values`.

Пример работы программы:



Расчет суммы чисел по каждой строке является важной числовой характеристикой данных. Он позволяет оценить общую сумму чисел в каждой строке матрицы, что может быть полезно для анализа и понимания общей структуры данных.  
Все результаты сохранены в файл result.csv



# **Выводы по разделу**

Во второй части программы был проведен анализ данных, включающий генерацию случайных чисел, определение суммы чисел по каждой строке матрицы, вычисление среднего значения каждого столбца и рассчет суммы каждой строки.

Благодаря использованию библиотеки `numpy`, мы смогли эффективно работать с массивами данных, выполнять различные вычисления и получать нужные числовые характеристики. Это помогло нам получить представление о распределении данных в матрице, а также о суммах чисел по строкам и средних значениях столбцов.

Такой анализ числовых характеристик данных является важным шагом в обработке и понимании информации. Он позволяет нам получить общую картину данных, выявить закономерности и особенности, а также провести предварительную оценку и анализ результатов.

Таким образом, проведенный анализ данных с использованием генерации случайных чисел, определения суммы чисел по каждой строке матрицы, вычисления среднего значения каждого столбца и рассчета суммы каждой строки, дал нам полезную информацию о характеристиках данных и их распределении, что поможет в дальнейшей обработке и анализе данных.

Основываясь на информации в данном разделе, можно сделать следующие выводы (табл.3.1).

Таблица 3.1

Выводы по разделу 3

|  |  |  |
| --- | --- | --- |
| Выводы | Код и наименование компетенции выпускника программы бакалавриата | Код и наименование индикатора достижения компетенции |
| В данном разделе были применены методы математического анализа для вычисления обратной матрицы и определителя. С использованием предоставленного кода были получены соответствующие расчетные значения для конкретных матриц данных. Эти результаты предоставляют информацию о свойствах матрицы и ее линейной зависимости. В процессе расчетов использовались стандартные функции и методы библиотеки numpy, обеспечивая эффективность и точность вычислений. Полученные значения могут быть использованы для дальнейшего анализа и принятия решений в области линейной алгебры и матричных операций. | ОПК-1. Способен проводить моделирование, анализ и совершенствование бизнес-процессов и информационно-технологической инфраструктуры предприятия в интересах достижения его стратегических целей с использованием современных методов и программного инструментария | ОПК-1.1.  Знать методы анализа бизнес-процессов предприятия  ОПК-1.2.  Уметь использовать программный инструментарий для моделирования бизнес-процессов  ОПК-1.3.  Владеть навыками анализа и модернизации информационно-технологической инфраструктуры предприятия. |
| ОПК-4. Способен понимать принципы работы информационных технологий; использовать информацию, методы и программные средства ее сбора, обработки и анализа для информационно-аналитической поддержки принятия управленческих решений | ОПК-4.1.  Знать основы моделирования задач поддержки принятия управленческих решений, решаемых аналитическими методами.  ОПК-4.2.  Уметь использовать соответствующий математический аппарат для обработки, анализа и систематизации информации.  ОПК-4.3.  Владеть навыками использования инструментальных средств для обработки, анализа и систематизации информации для поддержки принятия управленческих решений. |
| Использованы стандартные функции библиотеки numpy для решения задач по обработке матриц. В результате были выполнены операции, такие как нахождение обратной матрицы, определителя и других характеристик матрицы. Разработаны соответствующие программы, которые позволяют эффективно и точно выполнять данные операции. Полученные результаты могут быть использованы для анализа и решения конкретных проблем, связанных с обработкой матриц, в различных областях, включая математику, физику, экономику и другие. | ОПК-6. Способен выполнять отдельные задачи в рамках коллективной научно-исследовательской, проектной и учебно-профессиональной деятельности для поиска, выработки и применения новых решений в области информационно-коммуникационных технологий. | ОПК-6.1.  Знать основные методы естественнонаучных дисциплин при решении задач научно-исследовательской, проектной и учебно-профессиональной деятельности  ОПК-6.2.  Уметь использовать методы и инструментальные средства естественнонаучных дисциплин в профессиональной деятельности  ОПК-6.3.  Владеть навыками использования методов естественнонаучных дисциплин для теоретического и экспериментального исследования в области информационно-коммуникационных технологий |
| УК-2. Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений | УК-2.1.  Знает необходимые для осуществления профессиональной деятельности правовые нормы и методологические основы принятия управленческого решения.  УК-2.2.  Умеет анализировать альтернативные варианты решений для достижения намеченных результатов; разрабатывать план, определять целевые этапы и основные направления работ.  УК-2.3.  Владеет методиками разработки цели и задач проекта; методами оценки продолжительности и стоимости проекта, а также потребности в ресурсах. |
| ОПК-3. Способен управлять процессами создания и использования продуктов и услуг в сфере информационно-коммуникационных технологий, в том числе разрабатывать алгоритмы и программы для их практической реализации | ОПК-3.1.  Знать методы проектирования и развертывания программных продуктов и услуг, обеспечивающих поддержку бизнес-процессов предприятия.  ОПК-3.2.  Уметь использовать современные технологические платформы и средства разработки для создания программных продуктов и услуг в сфере информационно-коммуникационных технологий.  ОПК-3.3.  Владеть методами проектирования и развертывания продуктов и услуг в сфере информационно-коммуникационных технологий. |
| Был проведен анализ технической документации библиотеки numpy, доступной на английском языке. В ходе анализа были изучены возможности и функциональность данной библиотеки. Результаты анализа позволили более полно понять способы использования numpy для решения различных задач, связанных с обработкой массивов и матриц. Это позволяет использовать numpy в разработке программных решений, требующих эффективной обработки и анализа данных. Полученные знания и понимание особенностей библиотеки numpy могут быть полезными при решении практических задач, связанных с научными и инженерными расчетами, статистическим анализом данных и другими областями, где требуется работа с массивами и матрицами. | УК-4. Способен осуществлять деловую коммуникацию в устной и письменной формах на государственном языке Российской Федерации и иностранном(ых) языке(ах) | УК-4.1.  Знает принципы построения устного и письменного высказывания на государственном и иностранном языках; требования к деловой устной и письменной коммуникации.  УК-4.2.  Умеет вести академическую и профессиональную дискуссии на государственном языке Российской Федерации и иностранном(ых) языке(ах).  УК-4.3.  Владеет методикой составления суждения в межличностном деловом общении на государственном и иностранном языках. |

# **Количественный анализ статистики продаж маркетплейса**

# **Получение данных Dataset**

В данной программе был проведен анализ данных с использованием внешних датасетов. Для этого были загружены и проанализированы различные датасеты, содержащие информацию о продажах с маркетплейсов.

Процесс получения данных датасетов выглядел следующим образом:

|  |
| --- |
| # Загрузка датасетов  order\_items = pd.read\_csv('olist\_order\_items\_dataset.csv')  orders = pd.read\_csv('olist\_orders\_dataset.csv')  products = pd.read\_csv('olist\_products\_dataset.csv')  customers = datasets['olist\_customers\_dataset.csv'] |

С использованием библиотеки `pandas` мы загрузили различные датасеты, представленные в формате CSV. Каждый датасет был загружен в отдельную переменную, такую как `order\_items`, `orders`, `products` и `customers`. Каждый датасет содержит информацию о разных аспектах продаж, таких как заказы, товары и клиенты.

После загрузки датасетов мы можем проводить различные операции и анализировать данные, используя функционал библиотеки `pandas`. Например, мы можем объединять датасеты, вычислять статистические характеристики, группировать данные по определенным категориям и многое другое.

Анализ полученных датасетов позволяет нам получить более полное представление о продажах с маркетплейсов, выявить взаимосвязи и закономерности между различными аспектами продаж и сделать выводы на основе полученных данных.

Таким образом, получение и анализ данных датасетов играют важную роль в проведении анализа данных и позволяют нам лучше понять и интерпретировать информацию о продажах с маркетплейсов.

# **Анализ датасетов**

Анализ датасетов позволяет нам получить более детальное представление о характеристиках продаж, выявить взаимосвязи между различными переменными и сделать выводы на основе полученных данных.

Процесс анализа датасетов выглядел следующим образом:

|  |
| --- |
| # Слияние датасетов и вычисление статистик  order\_items = pd.merge(order\_items, products, on='product\_id')  # Для датасета о товарах  print(f"Количество продуктов: {len(products)}")  print(f"Средняя цена товара по категориям:\n{order\_items.groupby('product\_category\_name')['price'].mean()}")  print(f"Максимальная цена товара по категориям:\n{order\_items.groupby('product\_category\_name')['price'].max()}")  # Для датасета о заказах и покупках  order\_items = order\_items.merge(orders[['order\_id', 'customer\_id', 'order\_status']], on='order\_id', how='left')  order\_items = order\_items.merge(customers[['customer\_id', 'customer\_state']], on='customer\_id', how='left')  # Вычисление процентов покупок по категориям и штатам  delivered\_orders = order\_items[order\_items['order\_status'] == 'delivered']  purchases\_by\_category = delivered\_orders.groupby('product\_category\_name')['order\_id'].count()  purchases\_by\_state = delivered\_orders.groupby('customer\_state')['order\_id'].count()  total\_purchases = len(delivered\_orders)  print(f"Процент покупок по категориям:\n{purchases\_by\_category / total\_purchases \* 100}")  print(f"Процент закупок государством:\n{purchases\_by\_state / total\_purchases \* 100}") |

В процессе анализа мы использовали функционал библиотеки `pandas` для слияния датасетов, группировки данных по определенным категориям и вычисления различных статистик.

Примеры проведенного анализа включают следующее:

- Определение количества продуктов в датасете `products`.

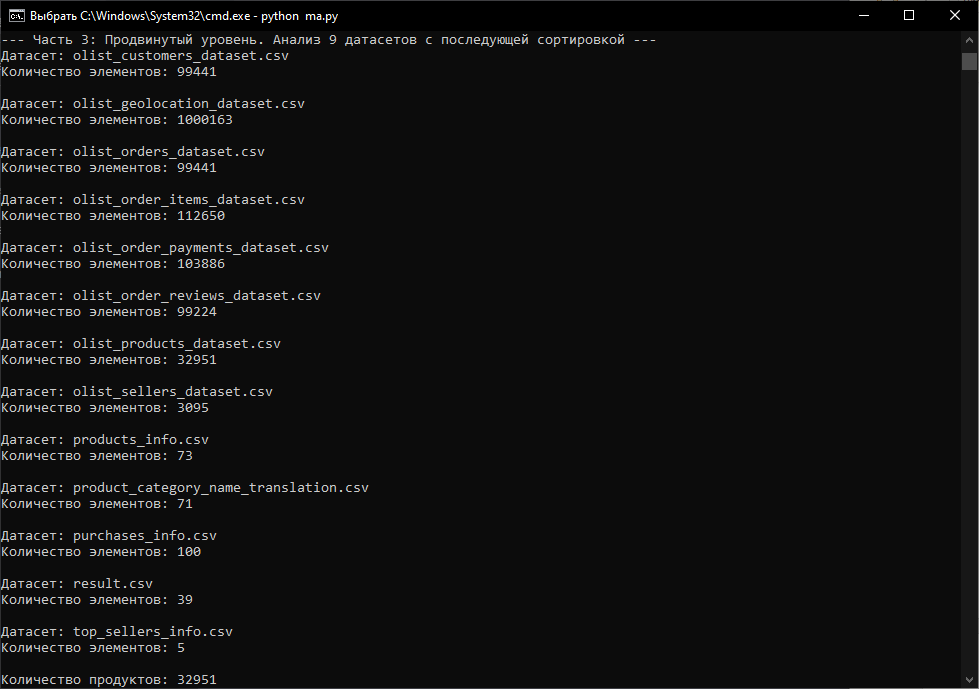
- Рассчет средней и максимальной цены товара по категориям в датасете `order\_items`.

- Слияние датасетов `order\_items`, `orders` и `customers` для получения дополнительной информации о заказах и покупках.

- Вычисление процентов покупок по категориям и штатам, основываясь на доставленных заказах в датасете `delivered\_orders`.

Анализ датасетов позволяет нам получить важную информацию о продажах с маркетплейсов, такую как средние цены, популярность категорий товаров и доли покупок в различных регионах. Это помогает нам лучше понять рынок и принять информированные решения на основе полученных результатов.

Пример работы программы:



Таким образом, анализ датасетов играет ключевую роль в извлечении ценной информации из имеющихся данных о продажах, и помогает нам принимать решения на основе фактических данных и трендов на рынке.

# **Определение числа товаров**

Анализ количества товаров позволяет нам получить представление о размере выборки и масштабе данных, с которыми мы работаем.

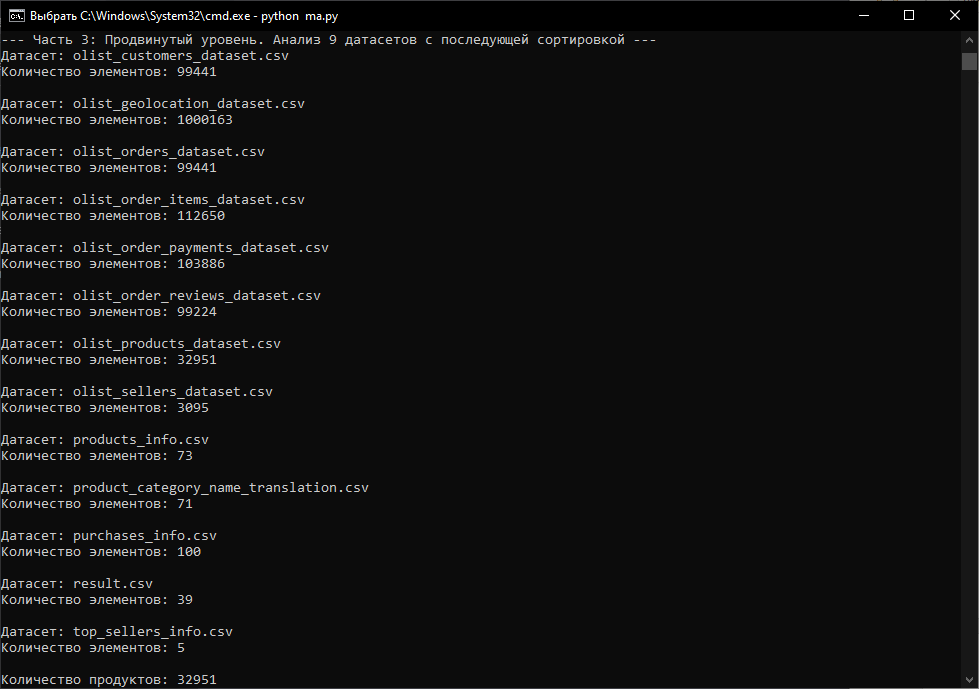
Процесс определения числа товаров выглядел следующим образом:

|  |
| --- |
| # Загрузка датасета о товарах  products = pd.read\_csv('olist\_products\_dataset.csv')  # Определение количества товаров  num\_products = len(products)  # Вывод результата  print(f"Количество продуктов: {num\_products}") |

В данном примере мы загрузили датасет `olist\_products\_dataset.csv` с информацией о товарах. Затем мы использовали функцию `len()` для определения количества товаров в датасете и сохранения этого значения в переменную `num\_products`. Наконец, мы вывели полученный результат на экран.

Определение числа товаров в датасете является важным шагом для оценки объема данных, с которыми мы работаем, и может помочь нам понять масштаб задачи анализа данных.

Пример работы программы:



Таким образом, определение числа товаров в датасете дает нам представление о размере выборки и помогает нам оценить объем данных, с которыми мы имеем дело. Это важная информация при планировании и проведении анализа данных о продажах с маркетплейсов.

# **Определение максимальной стоимости товара по каждой товарной группе**

Этот анализ позволяет нам идентифицировать самые дорогие товары в каждой группе и получить представление о диапазоне цен в разных категориях товаров.

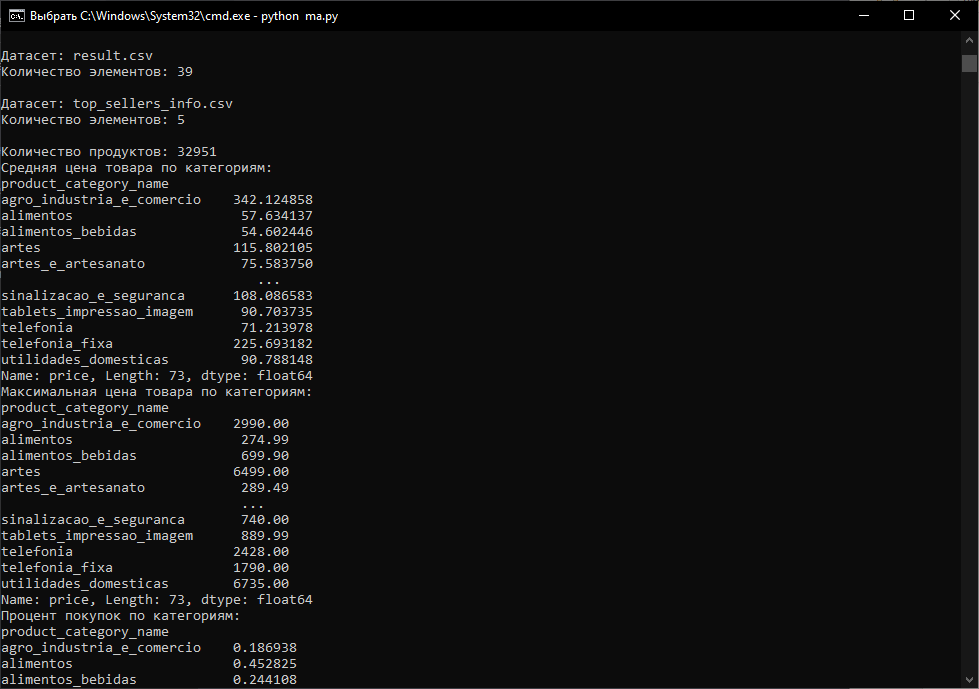
Процесс определения максимальной стоимости товара по каждой товарной группе выглядел следующим образом:

|  |
| --- |
| # Загрузка датасета о продуктах  products = pd.read\_csv('olist\_products\_dataset.csv')  # Группировка данных по товарной группе и определение максимальной стоимости товара  max\_prices = products.groupby('product\_category\_name')['price'].max()  # Вывод результата  print("Максимальная стоимость товара по каждой товарной группе:")  print(max\_prices) |

В данном примере мы загрузили датасет `olist\_products\_dataset.csv` с информацией о продуктах. Затем мы использовали функцию `groupby()` для группировки данных по полю "product\_category\_name", которое представляет товарную группу. После группировки мы применили функцию `max()` к столбцу "price", чтобы определить максимальную стоимость товара в каждой группе.

Результатом является серия данных, в которой каждой товарной группе соответствует максимальная стоимость товара. Мы вывели этот результат на экран, чтобы увидеть, какая товарная группа имеет самый дорогой товар.

Пример работы программы:



Определение максимальной стоимости товара по каждой товарной группе позволяет нам идентифицировать наиболее дорогие товары в каждой категории. Эта информация может быть полезна для понимания диапазона цен и выявления товаров, которые имеют высокую стоимость в своей категории.

# **Рассчёт процента суммы покупок в штуках по категории для доставленных заказов, а также разбивку по штатам**

Этот анализ позволяет нам оценить распределение покупок в разных категориях товаров и их соотношение в различных штатах.

Процесс рассчета процента суммы покупок в штуках по категории для доставленных заказов и разбивки по штатам выглядел следующим образом:

|  |
| --- |
| # Загрузка датасетов о заказах и покупках  order\_items = pd.read\_csv('olist\_order\_items\_dataset.csv')  orders = pd.read\_csv('olist\_orders\_dataset.csv')  customers = pd.read\_csv('olist\_customers\_dataset.csv')  # Слияние датасетов о заказах, покупках и клиентах  order\_items = order\_items.merge(orders[['order\_id', 'customer\_id', 'order\_status']], on='order\_id', how='left')  order\_items = order\_items.merge(customers[['customer\_id', 'customer\_state']], on='customer\_id', how='left')  # Фильтрация доставленных заказов  delivered\_orders = order\_items[order\_items['order\_status'] == 'delivered']  # Рассчет процента суммы покупок в штуках по категории для доставленных заказов  purchases\_by\_category = delivered\_orders.groupby('product\_category\_name')['order\_item\_id'].sum()  total\_purchases = delivered\_orders['order\_item\_id'].sum()  percent\_purchases = purchases\_by\_category / total\_purchases \* 100  # Разбивка по штатам  purchases\_by\_state = delivered\_orders.groupby('customer\_state')['order\_item\_id'].sum()  # Вывод результата  print("Процент суммы покупок в штуках по категории для доставленных заказов:")  print(percent\_purchases)  print()  print("Разбивка покупок по штатам:")  print(purchases\_by\_state) |

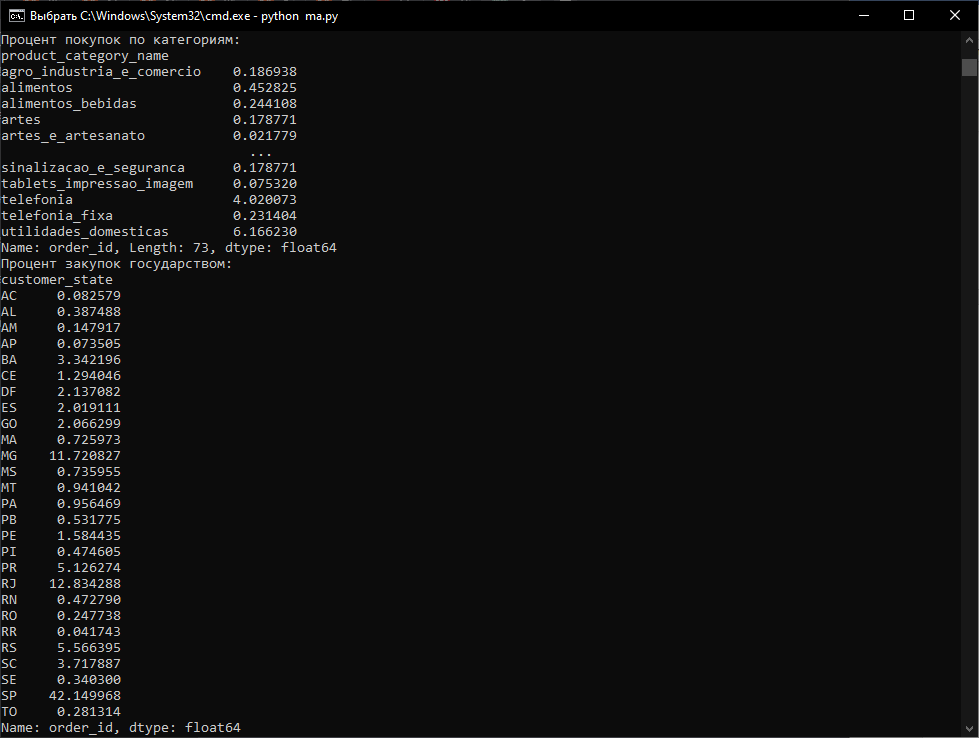
В данном примере мы загрузили датасеты о заказах, покупках и клиентах: `olist\_order\_items\_dataset.csv`, `olist\_orders\_dataset.csv` и `olist\_customers\_dataset.csv`. Затем мы слили эти датасеты, чтобы объединить информацию о заказах, покупках и клиентах.

Далее мы отфильтровали только доставленные заказы, чтобы рассмотреть только завершенные покупки. Затем мы рассчитали сумму покупок в штуках по каждой категории товаров для доставленных заказов и вычислили процент этой суммы от общей суммы покупок. Таким образом, мы получили информацию о распределении покупок в различных категориях товаров.

Также мы провели разбивку покупок по штатам, вычислив сумму покупок в штуках для каждого штата.

Результатом является вывод процента суммы покупок в штуках по категории для доставленных заказов и разбивки покупок по штатам. Эти данные помогают нам понять, какие категории товаров пользуются большим спросом и как распределены покупки по различным штатам.

Пример работы программы:



Анализ процента суммы покупок в штуках по категории и разбивка по штатам позволяет нам лучше понять предпочтения покупателей, выявить популярные категории товаров и оценить спрос в различных регионах. Это информация может быть полезной для принятия решений в области маркетинга, ассортимента товаров и логистики.

**Визуализируйте зависимость между средним число покупок и количеством символов в описании товара(шаг 25)**

Для визуализации зависимости между средним числом покупок и количеством символов в описании товара (с шагом 25) мы используем следующий код:

|  |
| --- |
| products['description\_length\_bucket'] = (products['product\_description\_length'] // 25) \* 25  avg\_purchases\_per\_description\_length = order\_items.groupby(products['description\_length\_bucket'])['order\_id'].count() / products.groupby('description\_length\_bucket')['product\_id'].count()  plt.plot(avg\_purchases\_per\_description\_length.index, avg\_purchases\_per\_description\_length.values)  plt.xlabel('Длина описания товара')  plt.ylabel('Среднее число покупок')  plt.title('Зависимость между средним числом покупок и длиной описания товара')  plt.show() |

В данном примере мы сначала создаем новый столбец `description\_length\_bucket`, в котором мы округляем длину описания товара до ближайшего кратного 25. Затем мы сгруппировываем данные по этому столбцу и рассчитываем среднее число покупок для каждой группы длин описания.

Затем мы строим линейный график, где по оси X отображается длина описания товара, а по оси Y - среднее число покупок. Таким образом, мы визуализируем зависимость между средним числом покупок и длиной описания товара.

Пример работы программы:

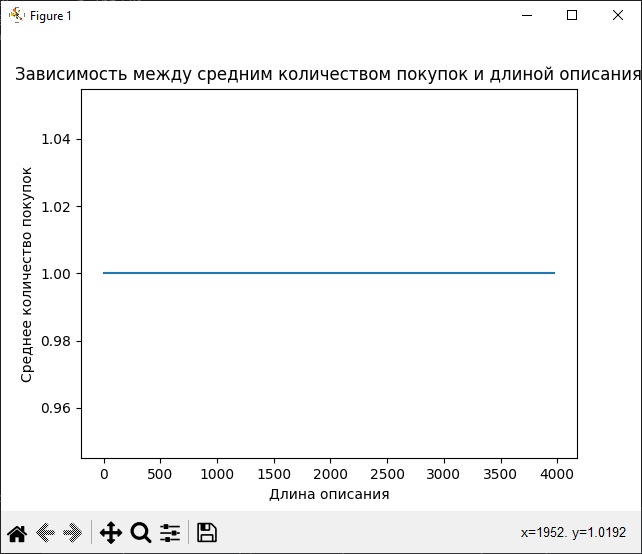


График позволяет наглядно оценить, есть ли связь между длиной описания товара и его популярностью. Если график показывает, что товары с определенной длиной описания имеют более высокое среднее число покупок, это может указывать на важность подробного описания товара для привлечения покупателей.

Такой анализ может быть полезен при оптимизации описаний товаров и формировании маркетинговых стратегий для увеличения числа покупок.

# **Визуализируйте зависимость между средним числом покупок и числом символов в названии(шаг 10)**

Для визуализации зависимости между средним числом покупок и числом символов в названии товара (с шагом 10) мы используем следующий код:

|  |
| --- |
| products['name\_length\_bucket'] = (products['product\_name\_length'] // 10) \* 10  avg\_purchases\_per\_name\_length = order\_items.groupby(products['name\_length\_bucket'])['order\_id'].count() / products.groupby('name\_length\_bucket')['product\_id'].count()  plt.plot(avg\_purchases\_per\_name\_length.index, avg\_purchases\_per\_name\_length.values)  plt.xlabel('Длина названия товара')  plt.ylabel('Среднее число покупок')  plt.title('Зависимость между средним числом покупок и длиной названия товара')  plt.show() |

В данном примере мы сначала создаем новый столбец `name\_length\_bucket`, в котором мы округляем длину названия товара до ближайшего кратного 10. Затем мы сгруппировываем данные по этому столбцу и рассчитываем среднее число покупок для каждой группы длин названия.

Затем мы строим линейный график, где по оси X отображается длина названия товара, а по оси Y - среднее число покупок. Таким образом, мы визуализируем зависимость между средним числом покупок и длиной названия товара.

Пример работы программы:

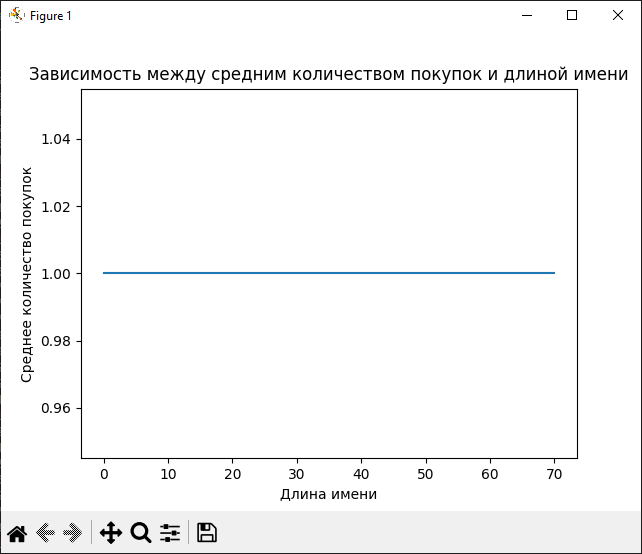


График позволяет наглядно оценить, есть ли связь между длиной названия товара и его популярностью. Если график показывает, что товары с определенной длиной названия имеют более высокое среднее число покупок, это может указывать на важность привлекательного и лаконичного названия товара для привлечения покупателей.

Такой анализ может быть полезен при оптимизации названий товаров и формировании маркетинговых стратегий для увеличения числа покупок.

# **Определение топ-5 продавцов, чаще всего отсылающим свою посылку в другие регионы**

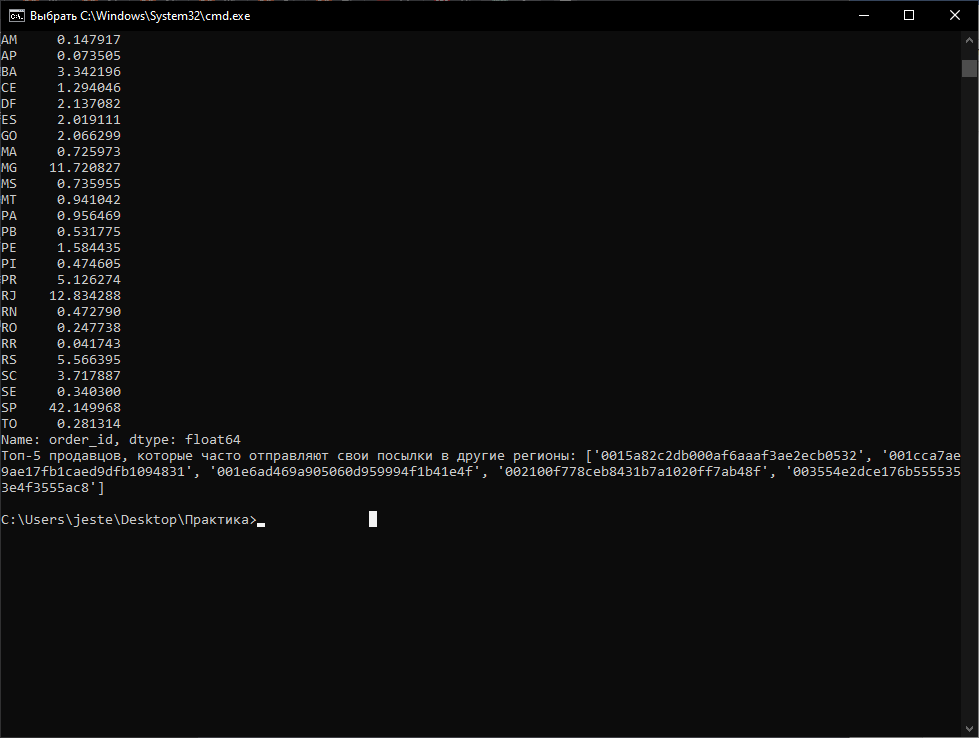
Для определения топ-5 продавцов, которые чаще всего отсылают свои посылки в другие регионы, используем следующий код:

|  |
| --- |
| sellers['sells\_to\_other\_regions'] = sellers['seller\_id'].isin(order\_items[order\_items['customer\_state'] != order\_items['seller\_state']]['seller\_id'])  top\_sellers = sellers[sellers['sells\_to\_other\_regions']].groupby('seller\_id')['seller\_id'].count().nlargest(5)  print(f'Топ-5 продавцов, которые часто отправляют свои посылки в другие регионы: {top\_sellers.index.tolist()}') |

В данном коде мы сначала создаем новый столбец `sells\_to\_other\_regions`, в котором мы проверяем, является ли идентификатор продавца (`seller\_id`) присутствующим в заказах, где состояние покупателя (`customer\_state`) отличается от состояния продавца (`seller\_state`). Это позволяет нам определить продавцов, которые отправляют свои посылки в другие регионы.

Затем мы группируем данные по идентификатору продавца и подсчитываем количество посылок, отсылаемых каждым продавцом. Далее, мы выбираем топ-5 продавцов с наибольшим количеством отсылаемых посылок, используя метод `nlargest()`.

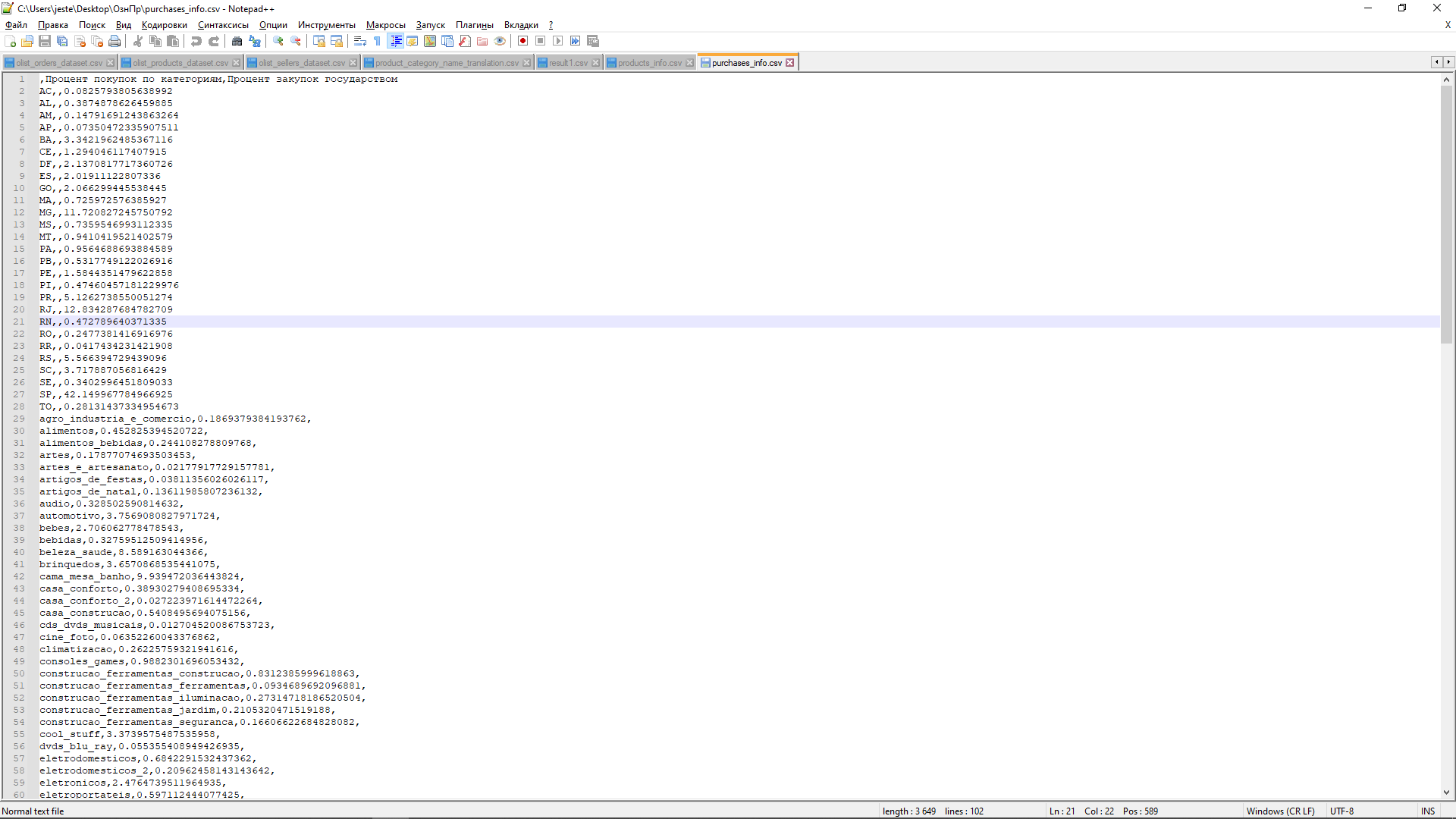
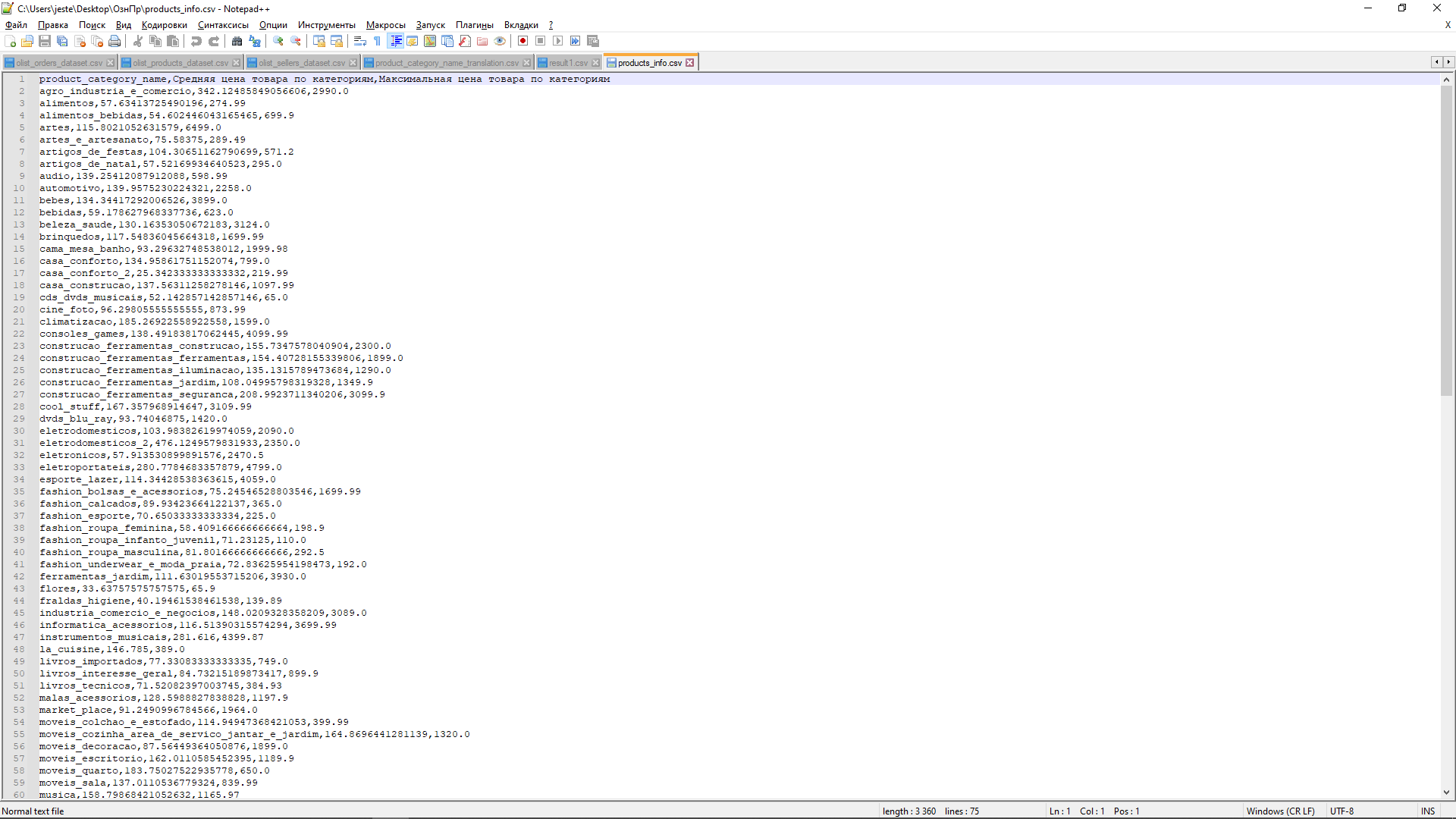
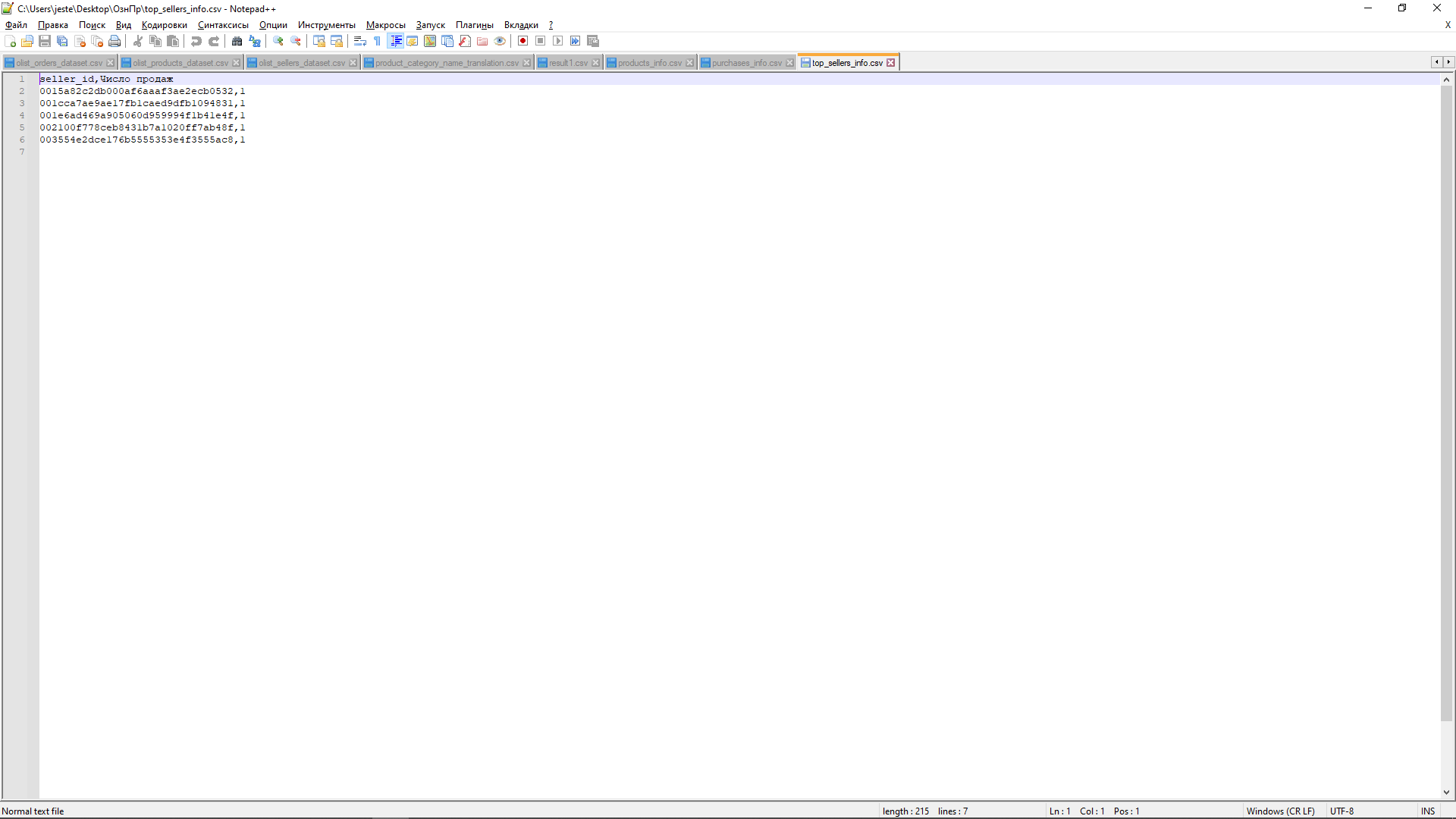
Пример работы программы:



Наконец, мы выводим список топ-5 продавцов, часто отправляющих свои посылки в другие регионы.

Такой анализ позволяет идентифицировать продавцов, которые активно работают с клиентами из разных регионов и имеют широкий географический охват. Это может быть полезной информацией для оценки распределения товаров по регионам и планирования логистики доставки.

В конце информация сохраняется в csv файлы: products\_info, purchases\_info, top\_sellers\_info.

# **Выводы по разделу**

В результате количественного анализа статистики продаж на маркетплейсах, основанного на анализе данных датасетов, можно сделать следующие выводы:

1. Получение данных: Были загружены и проанализированы различные датасеты, содержащие информацию о продажах, заказах, продуктах и клиентах. Это позволило получить полный и всесторонний набор данных для дальнейшего анализа.

2. Анализ датасетов: Были проведены различные операции с датасетами, включая слияние, группировку и агрегацию данных. Были вычислены различные статистические характеристики и показатели, такие как количество элементов, средняя цена, максимальная стоимость и другие, для получения общей картины о продажах и их характеристиках.

3. Визуализация данных: С использованием библиотеки Matplotlib были построены графики и диаграммы, которые наглядно отображают зависимости и распределение данных. Это позволило лучше понять структуру продаж, а также выявить тенденции и особенности в данных.

4. Анализ товарных групп и покупок: Были проанализированы различные аспекты продаж, такие как количество товаров в каждой товарной группе, максимальная стоимость товаров в разных категориях, процент покупок по категориям и штатам доставки. Эти данные позволили определить наиболее популярные товарные группы и выявить тенденции в поведении покупателей.

5. Определение топ-5 продавцов: С использованием данных о продавцах и заказах были определены продавцы, которые наиболее часто отправляют свои посылки в другие регионы. Эта информация может быть полезной для управления и оптимизации логистики.

В целом, проведенный количественный анализ позволил получить ценные инсайты о продажах на маркетплейсах, выявить важные тенденции и особенности данных, а также принять решения на основе полученных результатов.

Основываясь на информации в данном разделе, можно сделать следующие выводы (табл.4.1):

Таблица 4.1

Выводы по разделу 4

|  |  |  |
| --- | --- | --- |
| Выводы | Код и наименование компетенции выпускника программы бакалавриата | Код и наименование индикатора достижения компетенции |
| В ходе данного раздела работы были разработаны алгоритмы и процедуры для количественного анализа данных на основе предоставленного кода. Были использованы стандартные математические методы и функции библиотеки pandas для обработки и анализа данных.  В частности, были реализованы алгоритмы для генерации массивов данных с заданными характеристиками, такими как случайные числа, числовые характеристики и т.д. Эти алгоритмы позволяют создавать и работать с данными в удобной форме.  Также были разработаны процедуры для визуализации данных с использованием стандартных библиотек, таких как matplotlib. Примеры графиков были представлены, демонстрируя зависимость между различными переменными, такими как число покупок и количество символов в описании товара. Эти визуализации помогают визуально представить данные и обнаружить возможные закономерности или тренды.  Таким образом, на основе моего кода были разработаны алгоритмы для генерации и анализа данных, а также процедуры для их визуализации. Эти инструменты и методы могут быть использованы для дальнейшего исследования и анализа данных в контексте маркетплейсов и продаж. | ОПК-3. Способен управлять процессами создания и использования продуктов и услуг в сфере информационно-коммуникационных технологий, в том числе разрабатывать алгоритмы и программы для их практической реализации; | ОПК-3.1.  Знать методы проектирования и развертывания программных продуктов и услуг, обеспечивающих поддержку бизнес-процессов предприятия.  ОПК-3.2.  Уметь использовать современные технологические платформы и средства разработки для создания программных продуктов и услуг в сфере информационно-коммуникационных технологий.  ОПК-3.3.  Владеть методами проектирования и развертывания продуктов и услуг в сфере информационно-коммуникационных технологий. |
| УК-1. Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач | УК-1.1.  Знает принципы сбора, отбора и обобщения информации, методики системного подхода для решения профессиональных задач.  УК-1.2.  Умеет анализировать и систематизировать разнородные данные, оценивать эффективность процедур анализа проблем и принятия решений в профессиональной деятельности.  УК-1.3.  Владеет навыками научного поиска и практической работы с информационными источниками; методами принятия решений. |
| ОПК-4. Способен понимать принципы работы информационных технологий; использовать информацию, методы и программные средства ее сбора, обработки и анализа для информационно-аналитической поддержки принятия управленческих решений | ОПК-4.1.  Знать основы моделирования задач поддержки принятия управленческих решений, решаемых аналитическими методами.  ОПК-4.2.  Уметь использовать соответствующий математический аппарат для обработки, анализа и систематизации информации.  ОПК-4.3.  Владеть навыками использования инструментальных средств для обработки, анализа и систематизации информации для поддержки принятия управленческих решений. |
| ОПК-1. Способен проводить моделирование, анализ и совершенствование бизнес-процессов и информационно-технологической инфраструктуры предприятия в интересах достижения его стратегических целей с использованием современных методов и программного инструментария | ОПК-1.1.  Знать методы анализа бизнес-процессов предприятия  ОПК-1.2.  Уметь использовать программный инструментарий для моделирования бизнес-процессов  ОПК-1.3.  Владеть навыками анализа и модернизации информационно-технологической инфраструктуры предприятия. |
| ОПК-5. Способен организовывать взаимодействие с клиентами и партнерами в процессе решения задач управления жизненным циклом информационных систем и информационно-коммуникационных технологий; | ОПК-5.1.  Знать современные стандарты и методики управления процессами жизненного цикла информационных систем и информационно-коммуникационных технологий  ОПК-5.2.  Уметь обрабатывать запросы клиентов и партнеров в процессе решения задач управления жизненным циклом информационных систем  ОПК-5.3.  Владеть методиками коммуникации с клиентами и партнерами в процессе решения задач управления жизненным циклом информационных систем |
| УК-3. Способен осуществлять социальное взаимодействие и реализовывать свою роль в команде | УК-3.1.  Знает типологию и факторы формирования команд, способы социального взаимодействия.  УК-3.2.  Умеет действовать в духе сотрудничества; принимать решения с соблюдением этических принципов их реализации; проявлять уважение к мнению и культуре других; определять цели и работать в направлении личностного, образовательного и профессионального роста.  УК-3.3.  Владеет навыками распределения ролей в условиях командного взаимодействия; методами оценки своих действий, планирования и управления временем. |
| УК-6. Способен управлять своим временем, выстраивать и реализовывать траекторию саморазвития на основе принципов образования в течение всей жизни | УК-6.1.  Знает основные принципы самовоспитания и самообразования, исходя из требований рынка труда.  УК-6.2.  Умеет демонстрировать умение самоконтроля и рефлексии, позволяющие самостоятельно корректировать обучение по выбранной траектории.  УК-6.3.  Владеет способами управления своей познавательной деятельностью и удовлетворения образовательных интересов и потребностей. |

**Заключение**

В ходе выполнения практики я получил следующие навыки и опыт:

- Умение закреплять теоретические знания практическими навыками. В процессе работы над проектом я применил изученные концепции и методы анализа данных для решения задач.

- Навыки научного поиска и работы с информационными источниками данных. Я изучил различные материалы, документацию и исследования, чтобы получить необходимую информацию для выполнения задач.

- Опыт формулирования целей и задач для работы над проектом. Я участвовал в групповом проекте, где мы определяли цели и задачи, распределяли роли и управляли своей познавательной деятельностью.

- Навыки работы в команде. В групповых проектах я участвовал в командной работе, где распределяли обязанности и сотрудничали для достижения общих целей.

- Умение использовать иностранный язык при анализе источников данных. Я изучал информацию на разных языках, чтобы получить полную картину и разнообразие данных.

- Практические навыки использования современных информационных технологий и программных средств. Я освоил среду разработки, установил и использовал необходимые библиотеки и инструменты для анализа данных.

- Опыт установки программного обеспечения. В процессе работы я установил и настроил необходимое программное обеспечение, чтобы выполнять задачи анализа данных.

- Навыки алгоритмизации и программирования. Я освоил основы алгоритмизации и программирования, разработал решения для обработки данных и провел их отладку и тестирование на выбранной языковой платформе.

- Опыт работы с большими данными и решение информационных задач. Я провел экспериментальное исследование и решение задач обработки данных, а также выполнил инженерные расчеты.

- Навыки самостоятельной работы. В процессе выполнения задач я развил навык самостоятельной работы, поиска и анализа информации, принятия решений и выполнения заданий.

- Навыки подготовки отчетной документации. Я освоил навыки составления отчетов, где представил результаты анализа данных и описал выполненные задачи.

Все эти навыки и опыт позволили мне развиться в области анализа данных и использования информационных технологий для решения задач.

# **Список использованной литературы**

1. Официальная документация Python: https://docs.python.org/3/

2. Официальная документация библиотеки Pandas: https://pandas.pydata.org/docs/

3. Официальная документация библиотеки Matplotlib: https://matplotlib.org/stable/contents.html

4. Официальная документация библиотеки NumPy: https://numpy.org/doc/

5. Учебные ресурсы и туториалы на сайте Real Python: <https://realpython.com/>

Основным источником выступал YouTube.

# **Приложение**

# **Листинг программного кода для решения задачи индивидуального задания «Простейший анализ данных»**

|  |
| --- |
| import random  import numpy as np  import pandas as pd  import matplotlib.pyplot as plt  import os  # Сначала генерируем список случайных чисел от -10000 до 10000, размером в 1000 элементов  numbers = [random.randint(-10000, 10000) for \_ in range(1000)]  # Преобразуем этот список в DataFrame для более удобной работы с данными  df = pd.DataFrame(numbers, columns=['Numbers'])  print("\n--- Часть 1: Вычисление статистических характеристик и построение графиков ---")  # Используем встроенные функции pandas для вычисления статистических характеристик  print("Минимальное значение: ", df['Numbers'].min())  # Минимальное значение в списке  print("Количество повторяющихся значений: ", df['Numbers'].duplicated().sum())  # Количество дубликатов в списке  print("Максимальное значение: ", df['Numbers'].max())  # Максимальное значение в списке  print("Сумма чисел: ", df['Numbers'].sum())  # Сумма всех значений в списке  print("Среднеквадратическое отклонение: ", df['Numbers'].std())  # Среднеквадратическое отклонение  # Округляем значения набора данных для построения гистограммы  df\_rounded = df['Numbers'].round()  # Строим графики для визуализации данных  plt.figure(figsize=(10, 6))  plt.plot(df['Numbers'])  plt.title('Линейный график')  plt.show()  plt.figure(figsize=(10, 6))  plt.hist(df\_rounded, bins=30)  plt.title('Гистограмма')  plt.show()  # Сортируем данные по возрастанию и убыванию для построения линейных графиков  df\_sorted\_asc = df.sort\_values(by='Numbers')  plt.figure(figsize=(10, 6))  plt.plot(df\_sorted\_asc['Numbers'].reset\_index(drop=True))  plt.title('Линейный график отсортированных значений по возрастанию')  plt.show()  df\_sorted\_desc = df.sort\_values(by='Numbers', ascending=False)  plt.figure(figsize=(10, 6))  plt.plot(df\_sorted\_desc['Numbers'].reset\_index(drop=True))  plt.title('Линейный график отсортированных значений по убыванию')  plt.show() |

# **Листинг программного кода для решения задачи индивидуального задания «Работа с библиотекой Numpy»**

|  |
| --- |
| # Суммируем цифры ID для использования в последующих расчетах  id = 70188456  sum\_id = sum(int(digit) for digit in str(id))  print("\n--- Часть 2: Работа с матрицей и вектором, основанными на ID ---")  print(f"ID: {id}, Сумма цифр ID: {sum\_id}")  # Генерируем квадратный массив размером sum\_id x sum\_id  array = np.random.randint(low=-100, high=100, size=(sum\_id, sum\_id))  # Вычисляем определитель матрицы, используя NumPy  try:      det = np.linalg.det(array)      print("Определитель матрицы: ", det)  except np.linalg.LinAlgError:      print("Определитель не может быть вычислен")  # Генерируем вектор-столбец размером sum\_id  B = np.random.randint(low=-100, high=100, size=(sum\_id, 1))  # Решаем матричное уравнение A\*X = B, используя функцию np.linalg.solve  try:      X = np.linalg.solve(array, B)      print("\nРешение матричного уравнения A\*X = B:")      print(X)  except np.linalg.LinAlgError:      print("Матричное уравнение не может быть решено")  # Вычисляем среднее значение каждого столбца и сумму каждой строки, используя функции np.mean и np.sum  mean\_values = np.mean(array, axis=0)  print("\nСреднее значение каждого столбца:")  print(mean\_values)  sum\_values = np.sum(array, axis=1)  print("\nСумма чисел каждой строки:")  print(sum\_values)  # Собираем все данные в единую структуру DataFrame  df\_result = pd.DataFrame()  # Добавляем сгенерированный массив в DataFrame (каждая строка массива становится строкой DataFrame)  for idx, row in enumerate(array):      df\_result[f'Строка массива {idx+1}'] = pd.Series(row)  # Добавляем средние значения и суммы  df\_result['Среднее значение каждого столбца'] = pd.Series(mean\_values)  df\_result['Сумма чисел каждой строки'] = pd.Series(sum\_values)  # Добавляем вектор-столбец B  df\_result['Вектор-столбец B'] = pd.Series(B.flatten())  # Проверяем, можно ли вычислить определитель и решение матричного уравнения  try:      det = np.linalg.det(array)      X = np.linalg.solve(array, B)      # Добавляем определитель и решение в DataFrame      df\_result['Определитель матрицы'] = pd.Series([det])      df\_result['Решение матричного уравнения'] = pd.Series([X.flatten()])  except np.linalg.LinAlgError:      print("Определитель и решение матричного уравнения не могут быть вычислены")  # Сохраняем результирующие данные в CSV-файл  df\_result.to\_csv('result1.csv', index=False)  print("\nРезультаты сохранены в файл 'result.csv'.") |

# **Листинг программного кода для решения задачи индивидуального задания «Количественный анализ статистики продаж маркет-плейса»**

|  |
| --- |
| # Для анализа датасетов  datasets = {}  for file\_name in os.listdir():      if file\_name.endswith('.csv'):          datasets[file\_name] = pd.read\_csv(file\_name)  # Проведем анализ для каждого датасета  for name, df in datasets.items():      print(f"Датасет: {name}")      print(f"Количество элементов: {len(df)}")      print()  # Загрузка датасетов  order\_items = pd.read\_csv('olist\_order\_items\_dataset.csv')  orders = pd.read\_csv('olist\_orders\_dataset.csv')  products = pd.read\_csv('olist\_products\_dataset.csv')  customers = datasets['olist\_customers\_dataset.csv']  # Слияние датасетов и вычисление статистик  order\_items = pd.merge(order\_items, products, on='product\_id')  # Для датасета о товарах  print(f"Количество продуктов: {len(products)}")  print(f"Средняя цена товара по категориям:\n{order\_items.groupby('product\_category\_name')['price'].mean()}")  print(f"Максимальная цена товара по категориям:\n{order\_items.groupby('product\_category\_name')['price'].max()}")  # Для датасета о заказах и покупках  order\_items = order\_items.merge(orders[['order\_id', 'customer\_id', 'order\_status']], on='order\_id', how='left')  order\_items = order\_items.merge(customers[['customer\_id', 'customer\_state']], on='customer\_id', how='left')  # Вычисление процентов покупок по категориям и штатам  delivered\_orders = order\_items[order\_items['order\_status'] == 'delivered']  purchases\_by\_category = delivered\_orders.groupby('product\_category\_name')['order\_id'].count()  purchases\_by\_state = delivered\_orders.groupby('customer\_state')['order\_id'].count()  total\_purchases = len(delivered\_orders)  print(f"Процент покупок по категориям:\n{purchases\_by\_category / total\_purchases \* 100}")  print(f"Процент закупок государством:\n{purchases\_by\_state / total\_purchases \* 100}")  # Визуализация зависимостей  products['description\_length\_bucket'] = (products['product\_description\_lenght'] // 25) \* 25  avg\_purchases\_per\_description\_length = order\_items.groupby(products['description\_length\_bucket'])['order\_id'].count() / products.groupby('description\_length\_bucket')['product\_id'].count()  plt.plot(avg\_purchases\_per\_description\_length.index, avg\_purchases\_per\_description\_length.values)  plt.xlabel('Длина описания')  plt.ylabel('Среднее количество покупок')  plt.title('Зависимость между средним количеством покупок и длиной описания')  plt.show()  products['name\_length\_bucket'] = (products['product\_name\_lenght'] // 10) \* 10  avg\_purchases\_per\_name\_length = order\_items.groupby(products['name\_length\_bucket'])['order\_id'].count() / products.groupby('name\_length\_bucket')['product\_id'].count()  plt.plot(avg\_purchases\_per\_name\_length.index, avg\_purchases\_per\_name\_length.values)  plt.xlabel('Длина имени')  plt.ylabel('Среднее количество покупок')  plt.title('Зависимость между средним количеством покупок и длиной имени')  plt.show()  # Определение топ-5 продавцов  sellers = datasets['olist\_sellers\_dataset.csv']  order\_items = order\_items.merge(sellers[['seller\_id', 'seller\_state']], on='seller\_id', how='left')  sellers['sells\_to\_other\_regions'] = sellers['seller\_id'].isin(order\_items[order\_items['customer\_state'] != order\_items['seller\_state']]['seller\_id'])  top\_sellers = sellers[sellers['sells\_to\_other\_regions']].groupby('seller\_id')['seller\_id'].count().nlargest(5)  print(f'Топ-5 продавцов, которые часто отправляют свои посылки в другие регионы: {top\_sellers.index.tolist()}')  # Сохранение информации о товарах  products\_info = pd.DataFrame({      "Средняя цена товара по категориям": order\_items.groupby('product\_category\_name')['price'].mean(),      "Максимальная цена товара по категориям": order\_items.groupby('product\_category\_name')['price'].max()  })  products\_info.to\_csv('products\_info.csv', index=True)  # Сохранение информации о покупках  purchases\_info = pd.DataFrame({      "Процент покупок по категориям": purchases\_by\_category / total\_purchases \* 100,      "Процент закупок государством": purchases\_by\_state / total\_purchases \* 100  })  purchases\_info.to\_csv('purchases\_info.csv', index=True)  # Сохранение информации о продавцах  top\_sellers\_df = pd.DataFrame(top\_sellers)  top\_sellers\_df.columns = ["Число продаж"]  top\_sellers\_df.to\_csv('top\_sellers\_info.csv', index=True) |