

**Содержание**

[**Введение** 2](#_Toc137300352)

[ **Цель практики** 2](#_Toc137300353)

[**2. Основная часть** 3](#_Toc137300354)

[ **Выбор подходящей разработочной среды и необходимых библиотек** 3](#_Toc137300355)

[ **Объяснение причин выбора Python** 4](#_Toc137300356)

[ **Выбор инструментов для обработки и анализа** 4](#_Toc137300357)

[ **Сбор данных** 6](#_Toc137300358)

[ **Первичный анализ полученных данных** 7](#_Toc137300359)

[ **Определение формата для сохранения данных** 8](#_Toc137300360)

[ **Выводы по разделу** 9](#_Toc137300361)

[**3. Сортировка данных** 10](#_Toc137300362)

[ **Создание набора данных (Dataset)** 10](#_Toc137300363)

[ **Расчет основных статистических показателей для Series данных** 11](#_Toc137300364)

[ **Формирование Dataframe из Series данных и добавление дополнительных столбцов** 12](#_Toc137300365)

[ **Выводы по разделу** 13](#_Toc137300366)

[**4. Визуализация данных** 13](#_Toc137300367)

[ **Визуализация данных по итогу сортировки** 13](#_Toc137300368)

[ **Выводы по разделу** 14](#_Toc137300369)

[**5. Заключение** 15](#_Toc137300370)

[**6. Список использованной литературы** 15](#_Toc137300371)

**Введение**

Производственная практика (проектно-технологическая 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**

Целью данной практической работы является разработка программы на языке Python для сбора, обработки и анализа данных с веб-сайта. Для достижения этой цели были поставлены следующие задачи:

* Выбор подходящей платформы для разработки и необходимых библиотек.
* Сбор данных с веб-сайта.
* Предварительный анализ собранных данных.
* Выбор формата для хранения данных.
* Сортировка и анализ данных.
* Визуализация полученных результатов.

В качестве источника данных был выбран веб-сайт freetp.org, который содержит информацию о различных играх.

Моя личная цель в рамках этой практики - освоить веб-скрапинг и анализ данных. Веб-скрапинг — это ценный инструмент для извлечения информации из интернета, который может быть применен в различных областях, включая исследования, маркетинг и машинное обучение. Анализ данных, с другой стороны, является критическим навыком в нашем современном мире данных, который находит применение в самых разных сферах. Важность визуализации и работы с реальными данными также не может быть недооценена, поскольку они обеспечивают ценный практический опыт и способность эффективно передавать информацию другим. К тому-же проект подобного рода, может стать хорошим дополнением в будущее портфолио.

# **2. Основная часть**

# **Выбор подходящей разработочной среды и необходимых библиотек**

Для разработки программы была выбрана платформа python, так как она предоставляет широкий спектр библиотек для работы с данными и их визуализации,

а также имеет простой и понятный синтаксис. Пройдёмся подробнее по используемым библиотекам:

**csv**: Эта библиотека используется для чтения и записи файлов в формате CSV. В данном случае, она используется для сохранения собранных данных в файл.

**requests**: Библиотека requests используется для отправки HTTP-запросов. В данной программе она используется для получения веб-страниц.

**os**: Эта библиотека предоставляет функции для взаимодействия с операционной системой. В данной программе она используется для создания директорий и работы с файловой системой.

**time**: Библиотека time используется для работы со временем. В данной программе она используется для создания задержек.

**BeautifulSoup** (bs4): BeautifulSoup используется для парсинга HTML и XML документов. В данной программе она используется для извлечения данных из веб-страниц.

**docx**: Эта библиотека используется для создания, изменения и извлечения информации из файлов .docx. В данной программе она используется для создания отчета в формате Word.

docx2pdf: Эта библиотека используется для конвертации документов Word (.docx) в формат PDF. В данной программе она используется для создания отчета в формате PDF.

**pandas** (pd): Pandas - это библиотека для обработки и анализа данных. В данной программе она используется для обработки и анализа собранных данных.

**matplotlib**.pyplot (plt): Эта библиотека используется для создания статических, анимированных и интерактивных визуализаций в Python. В данной программе она используется для визуализации данных.

**logging**: Библиотека logging используется для логирования сообщений. В данной программе она используется для записи ошибок в файл.

**fuzzywuzzy** (process): FuzzyWuzzy - это библиотека для нечеткого сравнения строк. В данной программе она используется для объединения похожих категорий.

После выбора платформы и библиотек был проведен сбор данных с веб-сайта. Данные были собраны с помощью библиотеки requests, BS и сохранены в формате csv. word и pdf для удобства дальнейшей работы.

# **Объяснение причин выбора Python**

Python был выбран в качестве основного языка программирования для этой практики по следующим причинам:

**Простота** **и** **читаемость**: Python обладает простым и понятным синтаксисом, который упрощает чтение и понимание кода. Это делает Python отличным выбором для начинающих, так как он позволяет сосредоточиться на решении задачи, а не на понимании сложного синтаксиса.

**Широкий** **спектр** **библиотек**: Python имеет огромное количество библиотек для различных задач, включая обработку и анализ данных, машинное обучение, веб-скрапинг и многое другое. Это делает Python мощным инструментом для работы с данными.

**Поддержка** **сообщества**: Python имеет одно из самых больших и активных сообществ разработчиков, что обеспечивает быстрое решение проблем и постоянное обновление библиотек и инструментов.

**Портативность** **и** **межплатформенность**: Python является межплатформенным языком, что означает, что он может работать на различных операционных системах. Это делает его удобным для использования в различных средах и условиях.

**Интеграция** **с** **другими** **языками**: Python может быть легко интегрирован с другими языками программирования, такими как C или C++, что позволяет использовать преимущества этих языков при необходимости.

Все эти причины делают Python отличным выбором для работы с данными и веб-скрапинга в рамках этой практики.

# **Выбор инструментов для обработки и анализа**

Для обработки и анализа данных в этой практике были выбраны следующие инструменты:

1. **Pandas**: Это одна из самых популярных библиотек Python для обработки и анализа данных. Pandas предоставляет мощные и гибкие структуры данных (такие как DataFrame и Series), которые позволяют работать с различными типами данных и проводить сложные операции над ними. Библиотека поддерживает различные форматы данных, включая CSV, Excel и SQL, и имеет широкий спектр функций для манипулирования данными, таких как фильтрация, сортировка, агрегация и т.д.

2. **Matplotlib**: Это основная библиотека для визуализации данных в Python. Matplotlib позволяет создавать разнообразные графики и диаграммы, включая линейные графики, гистограммы, диаграммы рассеяния и многое другое. Это делает ее отличным инструментом для визуализации данных и представления результатов анализа в понятной и наглядной форме.

3. **BeautifulSoup**: Это библиотека Python для парсинга HTML и XML документов. BeautifulSoup используется для веб-скрапинга, т.е. извлечения данных из веб-страниц. Она предоставляет простые методы для поиска и навигации по DOM-структуре веб-страницы, что делает ее удобной для извлечения необходимой информации.

4. **Requests**: Это библиотека Python для отправки HTTP-запросов. Requests используется для получения веб-страниц, с которых затем извлекаются данные. Она обеспечивает простой и удобный интерфейс для работы с HTTP, поддерживает

множество методов запросов и позволяет управлять параметрами запроса, заголовками и т.д.

5. **FuzzyWuzzy**: Это библиотека Python для нечеткого сравнения строк. FuzzyWuzzy используется для объединения похожих категорий в данных, что помогает упростить анализ и улучшить качество результатов.

Все эти инструменты были выбраны из-за их функциональности, гибкости и простоты использования, а также широкой поддержки сообществом Python. Они обеспечивают

все необходимые средства для эффективной работы с данными в рамках этой практики.

# **Сбор данных**

Сбор данных в данной программе осуществляется с помощью библиотеки `requests` и `BeautifulSoup`.

Библиотека `requests` используется для отправки HTTP-запросов. В данной программе она используется для получения веб-страниц. Когда вы делаете запрос к веб-странице, сервер отвечает на запрос, возвращая HTML содержимое страницы.

Пример использования `requests` в коде:

1. Получение веб-страницы: В этом примере `requests.get()` используется для отправки GET-запроса к веб-странице. Заголовки передаются вместе с запросом для имитации действий браузера.

|  |
| --- |
| r = requests.get(base\_url + str(page\_number), headers=headers) |

2. Получение содержимого изображения: В этом примере `requests.get().content` используется для получения содержимого изображения, которое затем сохраняется в файл.

|  |
| --- |
| with open(image\_path, 'wb') as img\_file:  img\_file.write(requests.get('https://freetp.org' + image\_src).content) |

3. Получение веб-страницы игры: В этом примере `requests.get()` используется для получения веб-страницы конкретной игры.

|  |
| --- |
| game\_page = requests.get(link\_href, headers=headers) |

В каждом из этих случаев `requests.get()` отправляет HTTP GET-запрос к указанному URL и возвращает объект ответа, который используется для извлечения нужной информации.

Библиотека BeautifulSoup в Python используется для парсинга HTML и XML документов. Она создает дерево синтаксического анализа из страницы HTML, что облегчает извлечение данных из HTML-тегов.

Вот несколько примеров использования BeautifulSoup:

1. Парсинг HTML-страницы: В этом примере BeautifulSoup используется для парсинга HTML-страницы, полученной с помощью `requests.get()`.

|  |
| --- |
| html = BS(r.content, 'html.parser') |

2. Извлечение данных из HTML-тегов: В этом примере BeautifulSoup используется для извлечения данных из определенных HTML-тегов на веб-странице.

|  |
| --- |
| title = el.select('.header-h1 > a > h1')  link = el.select('.header-h1 > a')  image = el.select('.short-story .maincont div img')  description = el.select('.short-story .maincont div p')  comments = el.select('.mlink .argcoms a') |

3. Извлечение данных из веб-страницы игры: В этом примере BeautifulSoup используется для извлечения данных из веб-страницы конкретной игры.

|  |
| --- |
| game\_html = BS(game\_page.content, 'html.parser')  game\_mode = game\_html.select\_one('p:contains("Способ Игры:")') or game\_html.select\_one('p:contains("Способ игры:")')  game\_language = game\_html.select\_one('p:contains("Язык в Игре:")') or game\_html.select\_one('p:contains("Язык в игре:")')  game\_genre = game\_html.select\_one('p:contains("Жанр:")')  max\_players = game\_html.select\_one('p:contains("Максимальное количество игроков:")') or game\_html.select\_one('p:contains("Количество игроков:")')  single\_player = game\_html.select\_one('p:contains("Одиночная игра:")') |

В каждом из этих случаев BeautifulSoup анализирует HTML-страницу и позволяет извлекать данные из определенных HTML-тегов.

Таким образом, с помощью `requests` и `BeautifulSoup` можно собирать данные с веб-страниц и использовать их для дальнейшего анализа.

# **Первичный анализ полученных данных**

Первичный анализ данных включает в себя изучение основных характеристик данных, включая их размер, типы данных, пропущенные значения и базовые статистические показатели.

В моём коде первичный анализ данных проводится с помощью библиотеки pandas. Загружаются данные из файла CSV в DataFrame с помощью функции `pd.read\_csv()`. DataFrame — это двухмерная структура данных, предоставляемая pandas, которая позволяет проводить эффективный анализ данных.

|  |
| --- |
| df = pd.read\_csv(path + 'games.csv') |

Затем используется метод `describe()` для получения основных статистических характеристик каждого столбца DataFrame. Это включает в себя среднее значение, стандартное отклонение, минимальное и максимальное значения, а также квартили.

|  |
| --- |
| print(df.describe()) |

Этот метод полезен для получения быстрого представления о распределении данных в каждом столбце.

# **Определение формата для сохранения данных**

В данном коде используется несколько форматов для сохранения данных, каждый из которых служит своей цели.

CSV (Comma-Separated Values): используется CSV для сохранения собранных данных. Это универсальный формат, который позволяет хранить табличные данные в текстовом формате. CSV-файлы легко читаются и обрабатываются как людьми, так и машинами, и они совместимы с большинством систем обработки данных, включая pandas, который используется в коде. Вот фрагмент кода, отвечающий за сохранение данных в CSV-файл:

|  |
| --- |
| with open(path + 'games.csv', 'w', newline='', encoding='utf-8') as f:  writer = csv.writer(f)  writer.writerow(['Title', 'Link', 'Image', 'Description', 'Comments', 'Game Mode', 'Game Language', 'Game Genre', 'Max Players', 'Single Player'])  ...  writer.writerow([title\_text, link\_href, image\_src, description\_text, comments\_count, game\_mode\_text, game\_language\_text, game\_genre\_text, max\_players\_text, single\_player\_text]) |

DOCX (Word): также идёт сохранение данных в формате Word с помощью библиотеки `docx`. Это позволяет создавать структурированные документы с

текстом, изображениями и другими элементами. В моём коде используется `docx` для создания отчета, который содержит информацию о каждой игре:

|  |
| --- |
| doc = Document()  ...  doc.add\_heading(title\_text, level=1)  doc.add\_paragraph(f"Ссылка: {link\_href}")  doc.add\_paragraph(f"Описание: {description\_text}")  doc.add\_paragraph(f"Комментарии: {comments\_count}")  doc.add\_paragraph(f"Режим игры: {game\_mode\_text}")  doc.add\_paragraph(f"Языки: {game\_language\_text}")  doc.add\_paragraph(f"Жанры: {game\_genre\_text}")  doc.add\_paragraph(f"Количество игроков: {max\_players\_text}")  doc.add\_paragraph(f"Одиночная игра: {single\_player\_text}")  doc.add\_paragraph()  doc.add\_picture(image\_path)  ...  doc.save(path + 'games.docx') |

PDF: В самом конце документ Word конвертируется в формат PDF с помощью библиотеки `docx2pdf`. PDF-файлы удобны для распространения, поскольку они сохраняют форматирование и могут быть открыты на любом устройстве. Фрагмент, отвечающий за конвертацию в коде:

|  |
| --- |
| convert(path + 'games.docx', path + 'games.pdf') |

# **Выводы по разделу**

В разделе "Выбор платформы и данных" мы провели обширную подготовительную работу, которая обеспечила основу для успешного выполнения проекта.

Мы выбрали Visual Studio Code (VSCode) как нашу основную среду разработки из-за его удобства, гибкости и мощных инструментов для разработки на Python. Python был выбран в качестве языка программирования из-за его простоты, гибкости и мощного набора библиотек для работы с данными и веб-скрапинга.

Для сбора и анализа данных мы использовали набор инструментов, включая библиотеки Python, такие как BeautifulSoup и requests для веб-скрапинга, pandas для обработки и анализа данных, а также csv, docx и docx2pdf для сохранения и представления данных.

Мы успешно собрали данные с веб-сайта freetp.org, провели первичный анализ этих данных, и определили подходящие форматы для их сохранения, включая CSV для анализа данных, DOCX для создания структурированных отчетов и PDF для удобного распространения и просмотра.

В целом, этот раздел подчеркивает важность правильного выбора инструментов и подходов в начале проекта для обеспечения его эффективности и успешного выполнения.

|  |  |  |
| --- | --- | --- |
| Выводы | Сформированные компетенции | |
| Код | Содержание компетенции |
| Поиск, анализ и синтез информации был осуществлён. Системный подход применён. | УК-1 | Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач. |
| Были выбраны нужные библиотеки. | 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 |
| Научился осуществлять социальное взаимодействие и работать в команде. | УК-3 | Способен осуществлять социальное взаимодействие и реализовывать свою роль в команде |
| Смог сформировать отчёт в письменной форме на русском языке. | УК-4 | Способен осуществлять деловую коммуникацию в устной и письменной формах на государственном языке Российской Федерации и иностранном(ых) языке(ах) |
| Воспринял межкультурное разнообразие в отношении анализа данных. | УК-5 | Способен воспринимать межкультурное разнообразие общества в социально-историческом, этическом и философском контекстах |
| Смог спланировать свою работу по проекту и осуществить план. | УК-6 | Способен управлять своим временем, выстраивать и реализовывать траекторию саморазвития на основе принципов образования в течение всей жизни |
| Подготовил рабочее место. | УК-7 | Способен поддерживать должный уровень физической подготовленности для обеспечения полноценной социальной и профессиональной деятельности |
| Программа сохраняет условия безопасной жизнедеятельности и обеспечивает устойчивое развитие общество, не нанося вред экологии. | УК-8 | Способен создавать и поддерживать в повседневной жизни и в профессиональной деятельности безопасные условия жизнедеятельности для сохранения природной среды, обеспечения устойчивого развития общества, в том числе при угрозе и возникновении чрезвычайных ситуаций и военных конфликтов |
| Получил данные, соответствующие нужному уровню экономических решений. | УК-9 | Способен принимать обоснованные экономические решения в различных областях жизнедеятельности |
| Отчёт составлен без плагиата. | УК-10 | Способен формировать нетерпимое отношение к коррупционному поведению |
| Научился применять межпредметный знания на практике и при составление программы. | ОПК-1 | Способен применять естественнонаучные и общеинженерные знания, методы математического анализа и моделирования, теоретического и экспериментального исследования в профессиональной деятельности; |
| Показал уровень своего понимания в современных информационных технологиях при создании программы. | ОПК-2 | Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач профессиональной деятельности; |
| Ознакомился с библиотеками данных на основе библиографической культуры и учетом основных требований безопасности. | ОПК-3 | Способен решать стандартные задачи профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности; |
| Составил документацию по практике в форме отчётов. | ОПК-4 | Способен участвовать в разработке стандартов, норм и правил, а также технической документации, связанной с профессиональной деятельностью; |
| Установил нужные библиотеки для проекта. | ОПК-5 | Способен инсталлировать программное и аппаратное обеспечение для информационных и автоматизированных систем; |
| Осуществил подготовку к разработке, за счёт которой и осуществлял создание программы. | ОПК-6 | Способен анализировать и разрабатывать организационно-технические и экономические процессы с применением методов системного анализа и математического моделирования; |
| Сделал алгоритм для программы. | ОПК-7 | Способен разрабатывать алгоритмы и программы, пригодные для практического применения; |
| Способен в управлении над проектами подобного типа на всех стадиях жизненного цикла. | ОПК-8 | Способен принимать участие в управлении проектами создания информационных систем на стадиях жизненного цикла; |

# **3. Сортировка данных**

# **Создание набора данных (Dataset)**

Создание набора данных (Dataset) в данной программе происходит в процессе сбора данных с веб-сайта. Мой код использует библиотеку `csv` для создания файла `games.csv`, в который записываются данные, собранные с веб-страниц.

Вот фрагмент кода, который отвечает за создание набора данных:

|  |
| --- |
| with open(path + 'games.csv', 'w', newline='', encoding='utf-8') as f:  writer = csv.writer(f)  writer.writerow(['Title', 'Link', 'Image', 'Description', 'Comments', 'Game Mode', 'Game Language', 'Game Genre', 'Max Players', 'Single Player']) |

В этом фрагменте кода создается новый CSV-файл `games.csv`. Затем создается объект `writer`, который используется для записи данных в файл. С помощью метода `writerow()`, записывается первая строка файла, которая содержит названия столбцов.

Далее, в процессе обхода веб-страниц, собираются данные об играх и записываются в файл следующим образом:

|  |
| --- |
| writer.writerow([title\_text, link\_href, image\_src, description\_text, comments\_count, game\_mode\_text, game\_language\_text, game\_genre\_text, max\_players\_text, single\_player\_text]) |

В этом фрагменте кода метод `writerow()` используется для записи данных об одной игре в файл. Каждая игра записывается в отдельную строку файла.

Таким образом, в процессе работы программы создается набор данных, который сохраняется в файле `games.csv`.

# **Расчет основных статистических показателей для Series данных**

Библиотека pandas в Python предоставляет множество функций для расчета основных статистических показателей для Series данных. Вот некоторые из них:

1. `mean()`: Среднее значение

2. `median()`: Медиана

3. `mode()`: Мода

4. `min()`: Минимальное значение

5. `max()`: Максимальное значение

6. `sum()`: Сумма всех значений

7. `count()`: Количество непустых (не NaN) значений

8. `std()`: Стандартное отклонение

9. `var()`: Дисперсия

10. `quantile()`: Квантиль, значение в определенном проценте данных

В моей программе основные статистические показатели для Series данных рассчитываются с помощью метода `describe()` библиотеки pandas. Этот метод предоставляет быстрый обзор основных статистических показателей данных,

включая среднее значение, стандартное отклонение, минимальное и максимальное значения, медиану и квартили.

Вот фрагмент кода, где используется метод `describe()`:

|  |
| --- |
| df = pd.read\_csv(path + 'games.csv')  print(df.describe()) |

В этом примере сначала загружаются данные из CSV-файла в DataFrame с помощью метода `pd.read\_csv()`. Затем используется метод `describe()` для получения основных

статистических показателей для каждого столбца в DataFrame. Результат выводится на экран с помощью функции `print()`.

Метод `describe()` автоматически исключает столбцы, которые не являются числовыми, и вычисляет статистические показатели только для числовых столбцов.

# **Формирование Dataframe из Series данных и добавление дополнительных столбцов**

Создание DataFrame из Series данных и добавление дополнительных столбцов включает в себя несколько шагов. Я использовал библиотеку pandas для работы с данными в формате DataFrame.

Вот как это выглядит в коде:

Создание DataFrame из Series данных: В коде создаётся DataFrame из Series данных, используя функцию `pd.DataFrame()`. Это делается путем передачи Series данных в качестве аргумента функции, как показано ниже:

|  |
| --- |
| df = pd.DataFrame(game\_data) |

В этом примере `game\_data` — это Series данных, который нужно преобразовать в DataFrame.

Добавление дополнительных столбцов: Добавление новых столбцов в DataFrame в pandas можно сделать просто путем присвоения данных новому столбцу. Я добавил новые столбцы следующим образом:

|  |
| --- |
| df['Game'] = game\_name  df['Genre'] = game\_genre  df['Platform'] = game\_platform |

В этих примерах `Game`, `Genre` и `Platform` - это новые столбцы, которые я добавил в DataFrame `df`. `game\_name`, `game\_genre` и `game\_platform` — это данные, которые нужно добавить в эти столбцы.

# **Выводы по разделу**

В разделе "Сортировка информации" были выполнены ключевые этапы обработки и анализа данных. Сначала был создан набор данных (DataFrame) с использованием библиотеки pandas, что обеспечило удобство манипулирования данными. Затем были рассчитаны основные статистические показатели для каждой серии данных, что дало первоначальное представление о характере и распределении данных. Наконец, было выполнено преобразование Series данных в DataFrame и добавление дополнительных столбцов для более глубокого анализа. Все эти шаги обеспечили подробное понимание данных и подготовили их к дальнейшей визуализации и анализу.

|  |  |  |
| --- | --- | --- |
| Выводы | Сформированные компетенции | |
| Код | Содержание компетенции |
| Анализ данных был совершён. | УК-1 | Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач. |
| Способы анализа были выбраны. Анализ был выполнен. | 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 |
| Научился методам анализа и применил их на практике. | ОПК-1 | Способен применять естественнонаучные и общеинженерные знания, методы математического анализа и моделирования, теоретического и экспериментального исследования в профессиональной деятельности; |
| Показал понимания методов анализа. | ОПК-2 | Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач профессиональной деятельности; |
| Проверил и осуществил анализ данных в организационно-технических и экономических процессах. | ОПК-6 | Способен анализировать и разрабатывать организационно-технические и экономические процессы с применением методов системного анализа и математического моделирования; |
| Сделал алгоритм для программы. | ОПК-7 | Способен разрабатывать алгоритмы и программы, пригодные для практического применения; |
| Способен в управлении над проектами подобного типа на всех стадиях жизненного цикла. | ОПК-8 | Способен принимать участие в управлении проектами создания информационных систем на стадиях жизненного цикла; |

# **4. Визуализация данных**

# **Визуализация данных по итогу сортировки**

В моём коде используется библиотека `matplotlib.pyplot` для визуализации данных. Эта библиотека предоставляет множество функций для создания статических, анимированных и интерактивных визуализаций в Python.

Вот примеры использования `matplotlib.pyplot` в коде:

|  |
| --- |
| visualizations = {  'Количество игр в каждом жанре': {  'data': df['Game Genre'].value\_counts(),  'kind': 'pie'  },  'Количество игр в каждом игровом режиме': {  'data': df['Game Mode'].value\_counts(),  'kind': 'pie'  },  'Количество игр на каждом языке': {  'data': df['Game Language'].value\_counts(),  'kind': 'pie'  },  'Количество игр для каждого максимального значения игроков': {  'data': df['Max Players'].value\_counts(),  'kind': 'pie'  }  }  for i, viz in enumerate(visualizations.keys()):  print(f"{i+1}. {viz}")  viz\_choice = input("Введите номер визуализации (или нажмите Enter чтобы вернуться назад): ")  viz\_choice = int(viz\_choice) - 1  viz\_key = list(visualizations.keys())[viz\_choice]  viz = visualizations[viz\_key]  plt.figure(figsize=(10,6))  viz['data'].plot(kind=viz['kind'], autopct='%1.1f%%')  plt.title(viz\_key)  plt.ylabel('')  plt.show() |

В этом примере я создаю словарь `visualizations`, который содержит различные варианты визуализации данных. Затем предлагаю пользователю выбрать визуализацию, которую он хочет увидеть. Далее используя функцию `plot()` из `matplotlib.pyplot` для создания диаграммы, а также функции `title()`, `ylabel()`, и `show()` для установки заголовка, метки оси Y и отображения диаграммы соответственно, демонстрирую нужные графики.

# **Выводы по разделу**

В разделе визуализации данных были использованы различные методы для создания графиков и диаграмм, которые помогают лучше понять и интерпретировать собранные данные.

Мой код использовал библиотеку `matplotlib.pyplot` для создания визуализаций. Это включало в себя создание круговых диаграмм для представления количества игр в каждом жанре, игровом режиме, на каждом языке и для каждого максимального значения игроков. Эти визуализации были полезны для быстрого и наглядного представления распределения игр по различным категориям.

|  |  |  |
| --- | --- | --- |
| Выводы | Сформированные компетенции | |
| Код | Содержание компетенции |
| Осуществил визуализацию графиков с анализированным данными. | УК-1 | Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач. |
| Выбрал метод визуализации данных. | 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 |
| На практике применил полученные знания. | ОПК-1 | Способен применять естественнонаучные и общеинженерные знания, методы математического анализа и моделирования, теоретического и экспериментального исследования в профессиональной деятельности; |
| Продемонстрировал понимание изученных методик. | ОПК-2 | Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач профессиональной деятельности; |
| Правильно применил проанализированную информацию, осуществив её визуализацию. | ОПК-6 | Способен анализировать и разрабатывать организационно-технические и экономические процессы с применением методов системного анализа и математического моделирования; |
| Сделал алгоритм для программы. | ОПК-7 | Способен разрабатывать алгоритмы и программы, пригодные для практического применения; |
| Способен в управлении над проектами подобного типа на всех стадиях жизненного цикла. | ОПК-8 | Способен принимать участие в управлении проектами создания информационных систем на стадиях жизненного цикла; |

# **5. Заключение**

В ходе выполнения практической работы были успешно освоены и применены ключевые навыки в области веб-скрапинга, обработки и анализа данных, а также визуализации данных.

Использование языка Python и его библиотек (BeautifulSoup, pandas, matplotlib и других) позволило эффективно собирать, обрабатывать и анализировать данные с веб-сайта. Были изучены и применены методы для сбора данных с веб-страниц, их очистки и преобразования в удобный для анализа формат.

В процессе работы были получены важные навыки работы с данными, включая создание наборов данных, расчет основных статистических показателей, формирование DataFrame из данных Series и добавление дополнительных столбцов.

Визуализация данных была выполнена с использованием библиотеки matplotlib, что позволило наглядно представить результаты анализа.

В целом, практическая работа позволила углубить понимание важности и применимости навыков веб-скрапинга и анализа данных в современном мире.

В заключение, хотелось бы отметить, что выполнение этой практической работы было не только полезным, но и интересным опытом, который позволил применить и углубить теоретические знания на практике.

# **6. Список использованной литературы**

Python. Официальная документация. Url: <https://docs.python.org/>

Pandas. Официальная документация. Url: <https://pandas.pydata.org/>

Matplotlib. Официальная документация. Url: <https://matplotlib.org/>

Основным источником информации был YouTube.