

УТВЕРЖДЕНА

Решением Ученого совета
АНО ВО «Центральный университет»
«24» июня 2025 г.
Протокол № 2

**Рабочая программа дисциплины (модуля)
«Основы Python»**

Направление подготовки: 02.04.01 Математика и компьютерные науки

Направленность (профиль) подготовки: Машинное обучение

Квалификация (степень) выпускника: магистр

Форма обучения: очная

Срок освоения программы: 2 года

Год набора: 2025

**Москва
2025**

Содержание

| | |
|---|---|
| 1. Краткая характеристика дисциплины (модуля) | 3 |
| 2. Перечень планируемых результатов обучения..... | 5 |
| 3. Тематический план..... | 6 |
| 4. Содержание дисциплины (модуля)..... | 6 |
| 5. Учебно-методическое обеспечение | 7 |
| 6. Материально-техническое обеспечение | 7 |
| 7. Методические и оценочные материалы | 9 |

1. Краткая характеристика дисциплины (модуля)

Рабочая программа дисциплины (модуля) «Основы Python» составлена в соответствии с федеральным государственным образовательным стандартом высшего образования – магистратура по специальности 02.04.01 Математика и компьютерные науки, профиль Машинное обучение, утвержденный приказом Министерства науки и высшего образования Российской Федерации № 810 от 23.08.2017 года.

Изучение дисциплины (модуля) «Основы Python» помогает студентам освоить один из самых популярных и универсальных языков программирования, широко используемых в науке, бизнесе и IT-индустрии. Освоение основ Python позволяет эффективно автоматизировать задачи, анализировать данные и создавать программные продукты, что значительно расширяет профессиональные возможности.

Место дисциплины (модуля) в структуре образовательной программы

Настоящая дисциплина (модуль) включена в учебный план по программе подготовки магистратуры по направлению 02.04.01 Математика и компьютерные науки, профиль Машинное обучение и входит в обязательную часть Блока 1.

Дисциплина (модуль) изучается на 1 курсе в 1 семестре, проводится в формате онлайн-курса.

Цель изучения дисциплины (модуля): формирование фундаментальных знаний и навыков программирования на Python, позволяющих студентам решать базовые задачи с использованием коллекций, функций и объектно-ориентированного подхода, обеспечивая качественный и поддерживаемый код.

Задачи изучения дисциплины (модуля):

- освоить теоретические основы Python;
- приобрести практические навыки работы с данными и функциями;
- освоить обработку ошибок и объектно-ориентированное программирование;
- развить навыки реализации программ и организации кода;
- применять принципы алгоритмизации и качественной разработки.

В результате освоения дисциплины (модуля) обучающийся должен **знать:**

- основные принципы работы с переменными и их типами в Python;
- арифметические и логические операции, их синтаксис и применение;
- понятие функций, их назначение и синтаксис объявления;
- основные коллекции в Python: строки, списки, словари, множества и кортежи;
- принципы работы с файлами: чтение, запись, работа с текстовыми файлами;
- области видимости переменных в контексте функций и модулей;
- различия между процедурным и объектно-ориентированным подходами.

уметь:

- использовать встроенные методы и операции для работы со строками, списками, словарями, множествами и кортежами;
- писать и вызывать функции, включая функции с параметрами и возвратом значений;
- обрабатывать файлы: открывать, читать, записывать данные и управлять файлами через менеджеры контекста (with);
- выполнять базовую отладку кода и использовать рекомендации по написанию читаемого и поддерживаемого кода;
- создавать срезы для строк и списков, использовать итерации и циклы для работы с коллекциями;

- использовать основные конструкции Python для обработки ошибок (try-except);
- определять собственные классы, создавать экземпляры объектов, писать методы класса и использовать `__init__` для инициализации объектов.

владеть:

- навыками реализации программ для решения задач базового уровня сложности с использованием коллекций и функций;
- навыками организации кода в небольшие модули, используя функции для структурирования программы;
- навыками автоматизации обработки данных из файлов;
- навыками эффективного использования коллекции для хранения и обработки данных;
- навыками применения базовых принципов алгоритмизации при решении задач (например, работа с последовательностями, поиск, сортировка);
- навыками разработки решений, соответствующих требованиям к качеству кода (PEP 8);
- навыками реализации программ в объектно-ориентированной парадигме.

2. Перечень планируемых результатов обучения

Компетенции, формируемые в результате освоения дисциплины (модуля) при проведении учебных занятий в форме контактной работы обучающихся с педагогическими работниками Университета и в форме самостоятельной работы обучающихся:

| Компетенция | Содержание компетенции | Индикатор компетенции | Перечень планируемых результатов обучения по дисциплине (модулю) |
|-------------|---|-----------------------|--|
| ОПК-3. | Способен самостоятельно создавать прикладные программные средства на основе современных информационных технологий и сетевых ресурсов, в том числе отечественного производства | ОПК-3.1. | Знает основные принципы программирования, архитектуры программного обеспечения и современные языки программирования, а также особенности отечественных информационных технологий и сетевых ресурсов |
| | | ОПК-3.2. | Умеет разрабатывать прикладные программные средства, используя современные инструменты и технологии, а также интегрировать их с сетевыми ресурсами для решения конкретных задач |
| | | ОПК-3.3. | Имеет практический опыт разработки программных средств, используемых при построении математических моделей в естественных науках |
| ПК-6. | Способен разрабатывать программное обеспечение для решения прикладных задач в сфере машинного обучения | ПК-6.1. | Знает основные языки программирования, методы разработки программного обеспечения, а также принципы проектирования и архитектуры программных систем, применяемых в машинном обучении |
| | | ПК-6.2. | Умеет анализировать прикладные задачи, разрабатывать алгоритмы и реализовывать их в виде программного обеспечения, используя современные инструменты и технологии, а также проводить тестирование и отладку созданных решений |
| | | ПК-6.3. | Имеет практический опыт разработки программного обеспечения в рамках реальных проектов, включая участие в командах, где были успешно реализованы решения для конкретных прикладных задач в сфере профессиональной деятельности |

3. Тематический план

| № п/п | Наименование раздела дисциплины (модуля) | Трудоемкость, академические часы | | | | ТКУ (текущий контроль успеваемости) |
|--------|---|----------------------------------|-----------|----------|------------------------|--|
| | | <i>Очная форма</i> | | | | |
| | | Аудиторная работа | | Контроль | Самостоятельная работа | |
| Лекции | Консультации | | | | | |
| 1 | Синтаксис и базовые конструкции | | 1 | | 4 | Домашнее задание |
| 2 | Циклы и управление потоком исполнения | | 1 | | 4 | Домашнее задание Контрольная работа |
| 3 | Структуры данных | | 2 | | 4 | Домашнее задание |
| 4 | Функции и области видимости | | 2 | | 4 | Домашнее задание Контрольная работа |
| 5 | Работа с файлами и обработка ошибок | | 2 | | 4 | Домашнее задание Контрольная работа |
| 6 | Основы объектно-ориентированного программирования | | 2 | | 4 | Домашнее задание Проект |
| | <i>Зачет</i> | | | | 4 | |
| | <i>Итого:</i> | | 10 | | 4 | 24 |
| | Объем дисциплины (модуля) (в ак. ч.) | 38 | | | | |
| | Объем дисциплины (модуля) (в зач. ед.) | 1 | | | | |

4. Содержание дисциплины (модуля)

| №п/п | Наименование раздела дисциплины (модуля) | Содержание дисциплины (модуля) по темам |
|------|---|---|
| 1 | Синтаксис и базовые конструкции | Основы синтаксиса и типы данных Условные конструкции |
| 2 | Циклы и управление потоком исполнения | Циклы: for и while |
| 3 | Структуры данных | Списки, кортежи и строки Словари и множества |
| 4 | Функции и области видимости | Функции и области видимости |
| 5 | Работа с файлами и обработка ошибок | Работа с файлами и обработка ошибок |
| 6 | Основы объектно-ориентированного программирования | Объектно-ориентированное программирование |

5. Учебно-методическое обеспечение

Университет располагает полным набором лицензионного и свободно распространяемого программного обеспечения, включая продукты отечественного производства.

Каждый студент в течение всего периода обучения получает индивидуальный неограниченный доступ к электронно-библиотечной системе и электронной информационно-образовательной среде университета. Эти системы предоставляют возможность доступа к ресурсам из любой точки, где есть подключение к сети Интернет, как на территории университета, так и за его пределами.

Студентам обеспечен удаленный доступ к современным профессиональным базам данных и информационным справочным системам.

Основная литература:

1. Бизли, Д. Python. Исчерпывающее руководство : практическое руководство / Д. Бизли. - Санкт-Петербург : Питер, 2023. - 368 с. - (Для профессионалов). - ISBN 978-5-4461-1956-1. - Текст : электронный. - URL: <https://znanium.com/catalog/product/2123386>.

2. Яворски, М. Python. Лучшие практики и инструменты : практическое руководство / М. Яворски, Т. Зиаде. - Санкт-Петербург : Питер, 2021. - 560 с. - (Серия «Библиотека программиста»). - ISBN 978-5-4461-1589-1. - Текст : электронный. - URL: <https://znanium.com/catalog/product/1739603>.

Дополнительная литература:

1. Чернышев, С. А. Основы программирования на Python : учебник для вузов / С. А. Чернышев. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2025. — 349 с. — (Высшее образование). — ISBN 978-5-534-17139-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/567821>.

2. Нисчал, Н. Python — это просто. Пошаговое руководство по программированию и анализу данных : практическое руководство / Н. Нисчал. - Санкт-Петербург : БХВ-Петербург, 2022. - 416 с. - ISBN 978-5-9775-6849-4. - Текст : электронный. - URL: <https://znanium.ru/catalog/product/2139457>.

6. Материально-техническое обеспечение

Университет располагает материально-технической базой, соответствующей действующим противопожарным правилам и нормам и обеспечивающей проведение всех видов дисциплинарной и междисциплинарной подготовки, практической и научно-исследовательской работ обучающихся, предусмотренных учебным планом.

Помещения, которые представляют собой учебные аудитории для проведения занятий лекционного типа, занятий семинарского (практического) типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, а также помещения для самостоятельной работы и помещения для хранения и профилактического обслуживания учебного оборудования. Помещения укомплектованы специализированной мебелью и техническими средствами обучения, служащими для представления учебной информации большой аудитории.

Изучение дисциплины (модуля) обеспечивается в учебных аудиториях, оснащенных:

- столами и стульями;
- компьютерной техникой;
- механическими калькуляторами;
- специализированным оборудованием, включая демонстрационное оборудование.

Помещения для самостоятельной работы обучающихся, в том числе приспособленные для использования инвалидами и лицами с ограниченными возможностями здоровья,

оснащены компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду Университета.

Обучающимся предоставляется доступ (в том числе удаленный) к ресурсам информационно-телекоммуникационной сети «Интернет», электронным ресурсам (в том числе электронным библиотечным системам, современным профессиональным базам данных и информационным справочным системам):

| № | Наименование портала (издания, курса, документа) | Ссылка |
|----|--|---|
| 1 | Катастрофы, стихийные бедствия, аварии, эпидемии. Солнечная и геомагнитная активность. /ежедневный обзор | http://www.disasters.chat.ru |
| 2 | Каталог по безопасности жизнедеятельности | http://www.eun.chat.ru |
| 3 | Научная электронная библиотека eLibrary.ru библиотека | https://elibrary.ru/defaultx.asp |
| 4 | База данных для IT-специалистов | https://habr.com |
| 5 | База данных ScienceDirect | https://www.sciencedirect.com |
| 6 | Официальный сайт Министерства науки и высшего образования Российской Федерации | https://minobrnauki.gov.ru/ |
| 7 | Федеральный портал «Российское образование» | https://www.edu.ru/ |
| 8 | Информационная система "Единое окно доступа к образовательным ресурсам" | http://window.edu.ru/ |
| 9 | Единая коллекция цифровых образовательных ресурсов | http://school-collection.edu.ru/ |
| 10 | Федеральный центр информационно - образовательных ресурсов | http://fcior.edu.ru/ |

Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), в том числе комплект лицензионного программного обеспечения, современные профессиональные базы данных и информационные справочные системы:

| Наименование ПО | Производство | Лицензионное / свободно распространяемое |
|--|---------------|--|
| Операционные системы: | | |
| Microsoft Imagine (Windows Client, Server) | зарубежное | лицензионное |
| Браузеры: | | |
| Яндекс.Браузер | отечественное | свободно распространяемое |
| Google Chrome | зарубежное | свободно распространяемое |
| Офисные приложения: | | |
| Microsoft Imagine (Visio, OneNote) | зарубежное | лицензионное |
| TeXstudio | зарубежное | свободно распространяемое |
| Adobe Acrobat Reader | зарубежное | свободно распространяемое |
| Программное обеспечение для планирования и учета времени: | | |
| Toggle app | зарубежное | свободно распространяемое |
| Системы управления проектами: | | |
| Microsoft Imagine (Project) | зарубежное | лицензионное |
| Системы управления базами данных: | | |
| Microsoft Imagine (SQL Server) | зарубежное | лицензионное |
| Системы резервного копирования (backup): | | |
| Acronis Backup Advanced for HyperV | зарубежное | лицензионное |
| Справочно-правовые системы: | | |
| КонсультантПлюс: справочно-правовая система | отечественное | лицензионное |

| | | |
|---|---------------|---------------------------|
| Средства антивирусной защиты: | | |
| Kaspersky Endpoint Security для бизнеса Стандартный Russian Edition | отечественное | лицензионное |
| Пакеты программных средств и библиотек: | | |
| AutoPsy | зарубежное | свободно распространяемое |
| Interactive Disassembler (IDA) | зарубежное | свободно распространяемое |
| Системы управления библиографической информацией: | | |
| Zotero | зарубежное | свободно распространяемое |
| Сервисы и службы: | | |
| Bind | зарубежное | свободно распространяемое |
| Docker | зарубежное | свободно распространяемое |

7. Методические и оценочные материалы

Методические указания для обучающихся по освоению дисциплины (модуля)

В процессе изучения дисциплины (модуля) «Основы Python» в рамках текущего контроля успеваемости используются такие виды учебной работы, как консультации, контрольные работы, домашние задания, проект, а также различные виды самостоятельной работы обучающихся по заданию преподавателя, направленные на развитие навыков профессиональной лексики, закрепление практических профессиональных компетенций, поощрение инициатив.

Консультации – структурированные встречи, на которых преподаватели предоставляют индивидуальную или групповую помощь в освоении учебного материала, обсуждении вопросов и решении проблем, возникающих в процессе обучения.

Консультации могут включать разъяснение сложных тем, подготовку к экзаменам и помощь в выполнении проектных работ, что способствует более глубокому пониманию предмета и улучшению академической успеваемости.

Домашнее задание – набор задач по темам недели.

При работе над домашними заданиями важно внимательно ознакомиться с требованиями и сроками выполнения. Рекомендуется разбивать задания на этапы, чтобы избежать перегрузки и лучше усвоить материал. Использовать различные источники информации, включая учебники и онлайн-ресурсы, для более глубокого понимания темы.

Контрольная работа – письменная работа с набором задач, которые нужно решить за ограниченное время.

Цель контрольной работы – получить специальные знания по одной или нескольким темам дисциплины и продемонстрировать навыки их практического применения.

Проект – исследовательская работа по дисциплине (модулю) и презентация результатов.

Для успешной подготовки к проекту рекомендуется: четко определить цели и задачи проекта; составить план работы, разбив проект на этапы с указанием сроков выполнения каждого из них; использовать разнообразные источники информации и инструменты для исследования темы; регулярно проверять прогресс и вносить коррективы в план, если это необходимо.

Самостоятельная работа – работа студентов, направленная на углубленное изучение отдельных тем и вопросов учебной дисциплины (модуля).

В процессе самостоятельной работы студенты взаимодействуют с рекомендованными материалами при минимальном участии преподавателя. Задачи студента включают работу с конспектами лекций (обработка текста), повторное изучение учебных материалов планов

и тезисов ответов, изучение дополнительных тем, выполнение учебно-исследовательских заданий и другое.

Система оценивания результатов обучения по дисциплине (модулю)

Критерии получения уровня и оценивания сформированности компетенций по дисциплине (модулю) «Основы Python».

Оценивание уровня учебных достижений обучающихся по дисциплине (модулю) осуществляется в виде текущего контроля успеваемости.

Промежуточная аттестация по дисциплине (модулю) осуществляется в форме **зачета**, при этом проводится оценка компетенций, сформированных по дисциплине.

Для оценивания текущего контроля успеваемости и промежуточной аттестации используется десятибалльная шкала оценивания, которая соотносится с традиционной пятибалльной шкалой следующим образом:

| Десятибалльная оценка | Пятибалльная оценка | Оценка за зачет | Общая характеристика результата обучения по дисциплине (модулю) |
|-----------------------|---------------------|-----------------|--|
| 10 | Отлично | Зачтено | Студент полностью владеет знаниями, изложенными в рабочей программе, и глубоко осмысляет дисциплину (модуль). Он самостоятельно и логически последовательно отвечает на все вопросы, акцентируя внимание на наиболее важном. Умеет анализировать, сравнивать, классифицировать, обобщать, конкретизировать и систематизировать изученный материал, выделяя ключевые моменты и устанавливая причинно-следственные связи. Четко формулирует ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные задачи. Студент хорошо знаком с методами исследования, необходимыми для практической деятельности, и умеет связывать теоретические аспекты дисциплины (модуля) с практическими задачами. |
| 9 | Отлично | Зачтено | |
| 8 | Отлично | Зачтено | |
| 7 | Хорошо | Зачтено | Студент обладает знаниями предмета почти в полном объеме рабочей программы и самостоятельно, логически последовательно и всесторонне отвечает на все вопросы, акцентируя внимание на наиболее значимых моментах. Он умеет анализировать, сравнивать, классифицировать, обобщать, конкретизировать и систематизировать изученный материал, выделяя его ключевые аспекты и устанавливая причинно-следственные связи. |
| 6 | Хорошо | Зачтено | |

| Десятибалльная оценка | Пятибалльная оценка | Оценка за зачет | Общая характеристика результата обучения по дисциплине (модулю) |
|-----------------------|---------------------|-----------------|---|
| | | | Формулирует свои ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные ситуационные задачи. Студент хорошо знаком с методами исследования, необходимыми для практической деятельности, и умеет связывать теоретические аспекты предмета с практическими задачами. |
| 5 | Удовлетворительно | Зачтено | Студент обладает базовыми знаниями по дисциплине (модулю), но испытывает трудности при самостоятельных ответах и использует неточные формулировки. В ходе ответов он допускает ошибки, касающиеся сути вопросов. Студент способен решать только самые простые задачи и владеет лишь минимальным набором методов исследования. |
| 4 | Удовлетворительно | Зачтено | |
| 3 | Не сдан | Не зачтено | Студент не овладел обязательным минимумом знаний по предмету и не может ответить на вопросы, даже если преподаватель задает дополнительные наводящие вопросы. |
| 2 | Не сдан | Не зачтено | |
| 1 | Не сдан | Не зачтено | |

Зачет по дисциплине (модулю) «Основы Python» можно получить одним из способов:

- Решить 3 контрольные работы на 7 и более баллов.
- Решить 3 контрольные работы на 6 и более баллов и 2 домашних задания на 7 и более баллов.
- Решить 3 контрольные работы на 5 и более баллов и все домашние задания на 7 и более баллов.
- Решить проект на 7 и более баллов.

Текущий контроль успеваемости обучающихся по дисциплине (модулю)

Примерные домашние задания

Домашнее задание

1. Напишите краткий обзор (150-200 слов) о том, когда и почему был создан Python, а также о его основных особенностях.
2. Установите Python и настройте среду разработки (например, PyCharm или VS Code). Сделайте скриншоты процесса установки и настройки, а затем напишите краткое описание (50-100 слов) о том, как вы это сделали.
3. Напишите программу, которая запрашивает у пользователя его имя и возраст, а затем выводит сообщение в формате: "Привет, [имя]! Тебе [возраст] лет."
4. Создайте программу, которая запрашивает у пользователя два числа (целое и дробное), выполняет над ними арифметические операции (сложение, вычитание, умножение, деление) и выводит результаты.

5. Напишите программу, которая запрашивает у пользователя два булевых значения (например, True или False), и выводит результат логических операций (AND, OR, NOT) над ними.

Домашнее задание

1. Напишите программу, которая запрашивает у пользователя оценку (число от 1 до 100) и выводит соответствующий текст в зависимости от оценки (например, "Отлично", "Хорошо", "Удовлетворительно", "Неудовлетворительно").

2. Создайте программу, которая запрашивает у пользователя два числа и выводит, является ли первое число больше второго, меньше или равно, используя логические операторы.

3. Напишите программу, которая выводит все четные числа от 1 до 50, используя цикл for.

4. Создайте программу, которая запрашивает у пользователя ввод чисел и суммирует их, пока пользователь не введет 0. В конце выведите сумму.

5. Напишите программу, которая запрашивает у пользователя ввод чисел до тех пор, пока он не введет отрицательное число. Используйте оператор break для выхода из цикла.

Домашнее задание

1. Напишите функцию, которая принимает два числа и возвращает их произведение. Вызовите эту функцию и выведите результат.

2. Создайте функцию, которая принимает строку и число, а затем выводит строку заданное количество раз. Протестируйте функцию с различными аргументами.

3. Напишите программу, которая запрашивает у пользователя строку и выводит её в верхнем регистре, а также количество символов в строке.

4. Создайте список из 5 ваших любимых фруктов. Выведите его на экран и добавьте еще один фрукт в конец списка.

5. Напишите программу, которая создает список чисел, затем удаляет из него одно число по индексу, сортирует оставшиеся числа и выводит результат.

Примерные задания по контрольным работам

Контрольная работа 1.

1. Напишите программу на Python, которая выводит на экран строку "Hello, World!" и затем запрашивает у пользователя ввод его имени, после чего приветствует пользователя по имени. Объясните, какие типы данных используются в этой программе.

2. Создайте переменные разных типов данных (целое число, вещественное число, строка и булево значение) и выведите их на экран с помощью функции print(). Укажите, как проверить тип каждой переменной с использованием функции type().

3. Напишите код, который преобразует строку "123" в целое число, а затем умножает его на 2. Объясните, чем отличается явное преобразование типов от неявного.

4. Создайте список из пяти элементов разных типов (число, строка, булево). Попробуйте изменить один из элементов и выведите список до и после изменения. Объясните, почему список является изменяемым типом данных.

5. Напишите программу, которая запрашивает у пользователя два числа, складывает их и выводит результат. Добавьте обработку случая, если пользователь введет не число, с выводом сообщения об ошибке. Какие операторы и конструкции синтаксиса вы использовали?

Контрольная работа 2.

1. Напишите программу, которая запрашивает у пользователя число и проверяет, является ли оно положительным, отрицательным или нулем, выводя соответствующее сообщение. Используйте условные конструкции if, elif и else.

2. Создайте цикл while, который выводит числа от 1 до 10. Модифицируйте его так,

чтобы он останавливался, если число равно 5, используя оператор break.

3. Напишите программу с использованием цикла for, которая проходит по списку чисел [1, 2, 3, 4, 5] и выводит только четные числа. Объясните, как работает цикл for с итерируемыми объектами.

4. Создайте вложенные условные конструкции: программа должна проверить, является ли число больше 10, и если да, то проверить, делится ли оно на 2 без остатка. Выведите соответствующие сообщения для каждого случая.

5. Напишите цикл for, который выводит таблицу умножения для числа 5 (от 1 до 10). Добавьте условие, чтобы пропустить умножение на 3, используя continue. Объясните разницу между break и continue.

Контрольная работа 3.

1. Создайте список из 5 строк, добавьте к нему новый элемент и удалите первый. Затем преобразуйте список в кортеж и попробуйте изменить элемент кортежа. Объясните разницу между списками и кортежами.

2. Напишите программу, которая работает со строками: запросите у пользователя строку, выведите ее длину, переверните строку и замените все буквы "а" на "о". Какие методы строк вы использовали?

3. Создайте словарь с ключами "имя", "возраст", "город" и соответствующими значениями. Добавьте новый ключ "профессия", измените значение одного ключа и выведите все ключи и значения. Затем преобразуйте словарь в множество ключей и объясните, что такое множество.

4. Напишите функцию sum_list, которая принимает список чисел и возвращает их сумму. Вызовите эту функцию с разными списками и объясните концепцию областей видимости (локальные и глобальные переменные).

5. Напишите программу, которая открывает текстовый файл "example.txt" для записи, записывает в него строку "Hello, file!", затем читает содержимое файла и выводит его. Добавьте обработку ошибок на случай, если файл не найден, используя try и except. Объясните, почему важно обрабатывать ошибки при работе с файлами.

Примерные задания по проекту

1. Этап

ВАЖНО: не стоит тратить на это задание больше 30-40 минут. Постарайтесь придумать простое и устойчивое правило и быстро его реализовать.

Ваша первая задача по проекту – сформировать гипотезу для правила, которое могло бы отличить фродовые сообщения от «чистых».

Вход Десять реальных анонимизированных диалогов между покупателем и продавцом. 5 из них – фродовые, другие 5 – «чистые».

Выход Вы должны написать функцию, которая принимает на вход строку – первое сообщение продавца в чате – и присваивает ей один из двух лейблов: "fraud" или "clean".

Технические детали

- Строка, которая поступит на вход вашей функции, будет состоять из букв русского алфавита и всевозможных пунктуационных знаков.

- Строка, которая поступит вам на вход, будет сгенерирована из вероятностного распределения **на словах** и не будет иметь «физического» смысла. Каждое слово в строке будет сгенерировано независимо.

```
[ ] # пример строки, которая может поступить на вход вашей функции
fraud_example = ("карты но водителя заявку и не закончилась сообщений ждатель|"
                 " прошу отправителя Городе пытается Телеграм ее Получение"
                 " ответьте написать передадим дозвонились. заберёт не"
                 " транспортный дает получила~")

print(fraud_example)
```

Оценивание применяется функцию к тестовым данным и подсчитывается экономия маркетплейса от применения студентом правила. Оценка будет зависеть от финансовых показателей решения.

Часть 2.

Советы по выполнению задания

- графики должны быть понятными и приятными на вид. Обязательно подписывайте графики и подберите оптимальный размер, чтобы всё было видно
- часто используемый код оборачивайте в функции, так уменьшится вероятность ошибки и код станет красивее
- весь код, который вы написали, должен работать быстро. Иначе говоря, весь ноутбук должен исполняться не более чем 2-3 минуты.
- старайтесь не использовать магические константы. Например, если arr - массив чисел размера 1000, то для подсчёта среднего вместо

```
avg = sum(arr) / 1000 # плохо
```

пишите

```
avg = sum(arr) / len(arr) # хорошо
```

- **обязательно!** после выполнения задания нажмите кнопку *Перезапустить сеанс и выполнить весь код*, чтобы удостовериться, что ваше решение работает

Подсказка

В одном из заданий есть минорная ошибка и за её нахождение и исправление можно получить +10 баллов

Подготовка

Часть 1. Данные

Первым делом нам нужно научиться считывать тексты из файлов. Мы еще не разбирались, как это сделать в Python, но не переживайте – сейчас покажем.

Шаг 1 Загрузите данные

Шаг 1 Загрузите данные

```
[ ] !gdown 1UAs0dwBZ3gp9FnxrSu-kcUBBf-YtMVqr
!gdown 1B_eeBIavx9118fF2D0gzK0mEVu_9SM8i
```



Downloading...

From: <https://drive.google.com/uc?id=1UAs0dwBZ3gp9FnxrSu-kcUBBf-YtMVqr>

To: /content/data_clean.json

100% 994k/994k [00:00<00:00, 12.3MB/s]

Downloading...

From: https://drive.google.com/uc?id=1B_eeBIavx9118fF2D0gzK0mEVu_9SM8i

To: /content/data_fraud.json

100% 993k/993k [00:00<00:00, 9.33MB/s]

Шаг 2 Данные для этого проекта хранятся в формате json. С форматом можно познакомиться в документации, а пока можете просто считать данные, исполнив код ниже

```
[ ] import json

path_to_fraud = "/content/data_fraud.json"
path_to_clean = "/content/data_clean.json"

# считываем 1000 фродовых сообщений
with open(path_to_fraud, "r") as handler:
    fraud_messages = json.load(handler)

# считываем 1000 чистых сообщений
with open(path_to_clean, "r") as handler:
    clean_messages = json.load(handler)

# следующие две строки проверяют, что считанные списки сообщений имеют
# правильную длину
assert len(fraud_messages) == 1000
assert len(clean_messages) == 1000

print("Пример фродового сообщения:", fraud_messages[0])
print("Пример чистого сообщения:", clean_messages[0])
```

Если вы все сделали правильно, то ячейка выше должна отработать без ошибок и на экране должны появиться примеры двух сообщений. Не можете понять смысла сообщений? Ничего страшного, мы генерировали их из статистического распределения, поэтому «физического» смысла у сообщений нет. Но паттерны, которые встречаются во фродовых сообщениях, мы сохранили, поэтому исследование должно получиться интересным!

Важно: Все сообщения состоят из букв русского алфавита и всевозможных пунктуационных знаков.

Константы

| КОНСТАНТА | ОПИСАНИЕ | ЗНАЧЕНИЕ |
|---------------------|--|----------------|
| DAILY_PURCHASES | Среднее количество покупок, совершаемых на маркетплейсе каждый день. По каждой покупке создается чат Покупатель-Продавец | 200 000 |
| FRAUD_SHARE | Доля покупок, которые приходятся на продавцов-мошенников | 5% |
| FALSE_POSITIVE_COST | Цена одной ошибки FALSE POSITIVE (заблокировали честное сообщение), в рублях | 10 000 |
| FALSE_NEGATIVE_COST | Цена одной ошибки FALSE NEGATIVE (пропустили фродерское сообщение), в рублях | 75 000 |

Бейзлайн и оценка качества

Прежде чем начать анализ важно зафиксировать бейзлайн – простое решение, относительно которого можно измерять эффективность своей работы.

Задача 1. Бейзлайн В качестве бейзлайна реализуйте три бизнес-правила:

- constant_fraud --- правило, которое классифицирует каждое сообщение как фрод
- constant_clean --- правило, которое классифицирует каждое сообщение как чистое
- first_hypothesis --- правило, которое вы придумали в первом шаге работы над проектом

Каждое бизнес-правило принимает на вход сообщение в виде строки и возвращает его класс: «fraud» или «clean».

Задача 2. Оценка качества [10 баллов] Бейзлайны готовы --- теперь применим их к данным проекта и оценим их качество в деньгах.

2.1. Параметры Чтобы подсчитать нашу целевую метрику (дневные потери маркетплейса), нам нужно научиться вычислять два параметра:

- `false_positive_rate`: доля ошибок типа FALSE POSITIVE (от 0 до 1)
- `false_negative_rate`: доля ошибок типа FALSE NEGATIVE (от 0 до 1)

Ваша задача:

1. Примените каждый из трех бейзлайнов к данным (`fraud_messages` и `clean_messages`)
2. По результатам предсказаний определите `false_positive_rate` и `false_negative_rate` каждого бейзлайна
3. Выведите результаты на экран с точностью до 3 знаков после запятой (используйте функцию `round()`)

2.2. Деньги Теперь нужно перевести значения FPR и FNR в деньги. Напишите функцию, которая принимает на вход значения этих параметров и вычисляет ежедневные потери маркетплейса при внедрении каждого бейзлайна. Примените функцию к результатам задачи 2.1, чтобы сравнить три бейзлайна.

Частотный анализ

Итак, мы подготовили бейзлайны --- пришло время провести аналитику и построить по-настоящему хорошее бизнес-правило. Первый шаг --- подготовка данных и визуальный анализ.

Задача 3. Облако слов Давайте начнем с того, что посмотрим на облака слов фродовых и чистых сообщений, чтобы составить первое впечатление о данных. Необходимый для этой задачи инструментарий можно посмотреть в разделе учебника по визуализации.

3.1 Подготовка Наши сообщения состоят из букв русского алфавита и знаков препинания. Буквы составляют слова, а вот знаки препинания особенно ценной информации для анализа не несут (а еще могут использоваться фродерами для маскировки слов, а-ля Т-е-Л-е-г-р-@м). Прежде чем строить облака слов, проведите первичную обработку сообщений:

- удалите знаки препинания
- приведите все слова к нижнему регистру

2.3. Анализ Проанализируйте результаты 2.1 и 2.2 и объясните их с точки зрения бизнеса.

- Почему помечать все сообщения как чистые более выгодно, чем все как фрод?
- Как ваша собственная гипотеза соотносится с простыми константными правилами?

Как вам удалось их превзойти / почему не удалось?

Частотное бизнес-правило [20 баллов] Давайте углубимся в идею с ключевыми словами и построим бизнес-правило, которое блокирует сообщения с определенными ключевыми словами.

Анализ Чтобы реализовать такое бизнес-правило, проделайте следующие шаги:

- Составьте список слов, которые встречаются в нормализованных сообщениях хотя бы один раз (общий список для фродовых и чистых сообщений). Здесь хорошо использовать структуру данных множество: `set` (пройдем на 4 неделе курса).

- Для каждого слова представьте правило, которое блокирует все сообщения с этим словом. Определите `false_positive_rate` и `false_negative_rate` такого правила

- Для каждого слова определите метрику эффективности (в деньгах) соответствующего правила (с шага 2)

- Отсортируйте все слова по эффективности соответствующего правила (от самых эффективных к самым неэффективным) и постройте `bar chart` финансовых результатов для первых 20 слов

- Из графика определите список ключевых слов, по которому ваше правило будет блокировать сообщение. Ваше правило должно блокировать сообщение если в нем есть хотя бы одно ключевое слово.

Разработка Теперь реализуйте ваше правило end-to-end:

- *Вход*: необработанное сообщение (из сырых данных)
- *Внутренность*: Подготовка и нормализация текста (задачи 3.1 и 4), поиск ключевых слов (задача 5.1)
- *Выход*: Вердикт («fraud» или «clean»)
- **Оценка эффективности** Оцените эффективность вашего итогового бизнес-правила (аналогично задачам 2.1 и 2.2)
 - Если всё реализовано корректно, у вас должно получиться не более 420 млн дневных потерь
 - Придумайте end-to-end бизнес-правило на основе наивного байеса. Ваше правило должно принимать на вход сообщение из сырых данных и выдавать вердикт («fraud» или «clean»). **Нельзя** использовать готовую реализацию, нужно написать свой классификатор.
 - Оцените его эффективность
 - Прокомментируйте результаты (Как результаты соотносятся с частотным правилом?, Как можно дальше развивать ваше решение?)

Задания для промежуточной аттестации по дисциплине (модулю)

| № п/п | Задание | Ответ | Компетенция |
|-------|--|-------------------|-------------|
| 1. | Какой результат выполнения следующего кода? <pre>x = 0 x += 1 x += 2 print(x)</pre> | 3 | ОПК-3 |
| 2. | Какой результат выполнения следующего кода? <pre>print("Hello".upper())</pre> | HELLO | ОПК-3 |
| 3. | Какой результат выполнения следующего кода? <pre>a = [1, 2, 3, 2, 4] print(a.count(2))</pre> | 2 | ОПК-3 |
| 4. | Какой результат выполнения следующего кода? <pre>a = [10, 20, 30, 40] print(a[1:3])</pre> | [20, 30]/ [20,30] | ОПК-3 |
| 5. | Какой оператор используется для передачи переменного числа позиционных аргументов в функцию? А. *args В.%args С. \$args D. <args | А | ОПК-3 |
| 6. | Какой метод используется для чтения всего содержимого файла в виде списка строк? А. detect() В. writelines() | С | ОПК-3 |

| | | | |
|-----|---|-----------------------------|------|
| | C. readlines() D. wrote() | | |
| 7. | Сколько раз выполнится следующий цикл? <pre>for i in range(5, 15, 5): print(i)</pre> | 2 | ПК-6 |
| 8. | Какой результат выполнения следующего кода? <pre>print(10 // 3)</pre> | 3 | ПК-6 |
| 9. | Какой результат выполнения следующего кода? <pre>x = 8 y = 2 print(x / y)</pre> | 4.0 / 4,0 | ПК-6 |
| 10. | Какой результат выполнения следующего кода? <pre>print(2 ** 3)</pre> | 8 | ПК-6 |
| 11. | Какая функция возвращает длину списка? A. sum() B. len() C. list() D. count() | B | ПК-6 |
| 12. | Ниже приведён набор строк кода. Выбери те строки, которые необходимы для создания и вызова функции greet(name), выводящей приветствие "Привет, Мир!". Варианты: A. def greet(name): B. print(f"Привет, {name}!") C. return "Мир" D. greet("Мир") E. print("Привет, Студент!") | ABD/ADB/BAD/B DA/DBA/DAB | ПК-6 |