**Отчет по самостоятельной работе №6**

**по дисциплине МДК 01.02 “Инструментальные средства разработки программного обеспечения”.**

Выполнил: студент

группы 319

Попова Маргарита Вячеславовна

Дата 26.11.2024

**Цель работы**

Изучить методы и подходы к обработке исключений, научиться применять конструкции try-catch, обрабатывать и регистрировать исключения для повышения стабильности программного обеспечения.

**Основная структура задания**

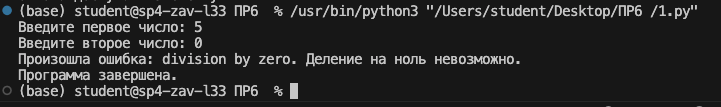
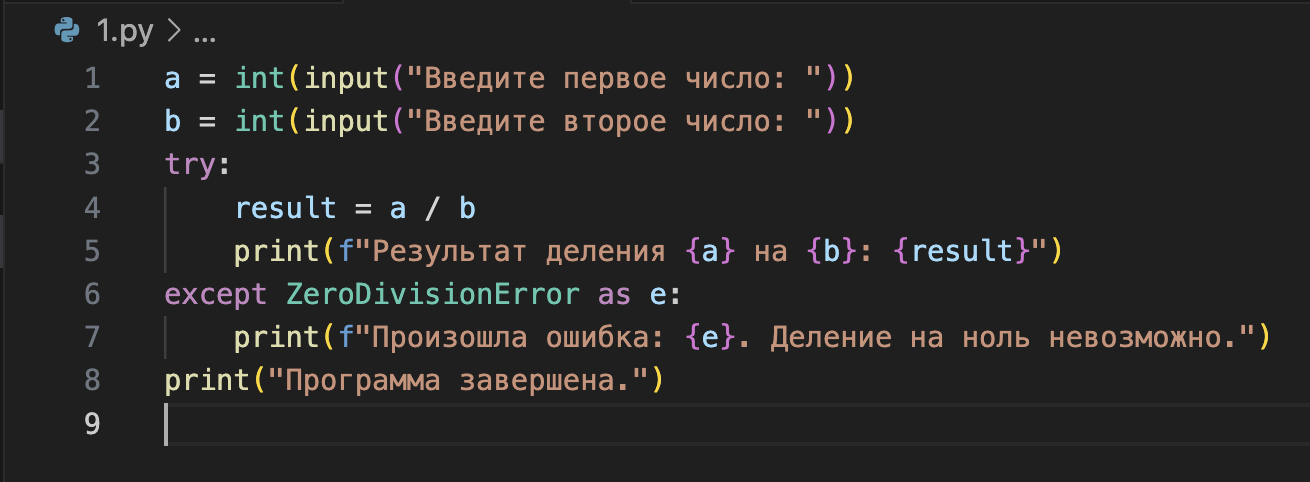
Цель данной работы –

Практическая часть

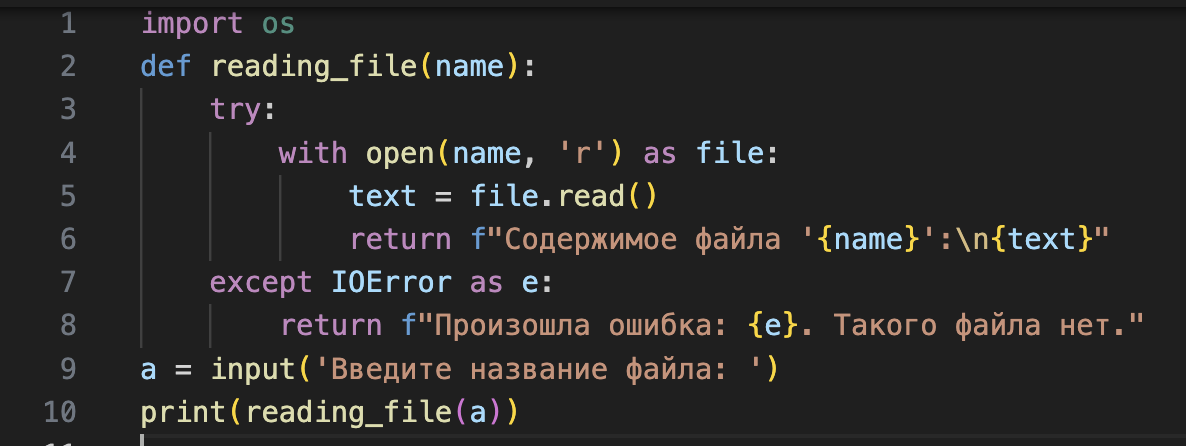
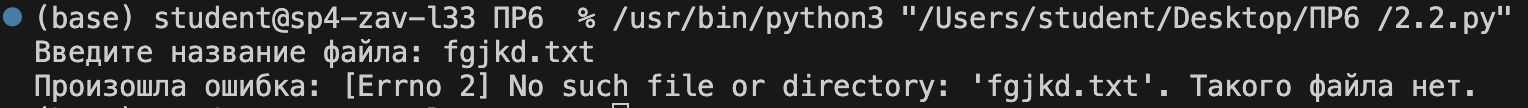
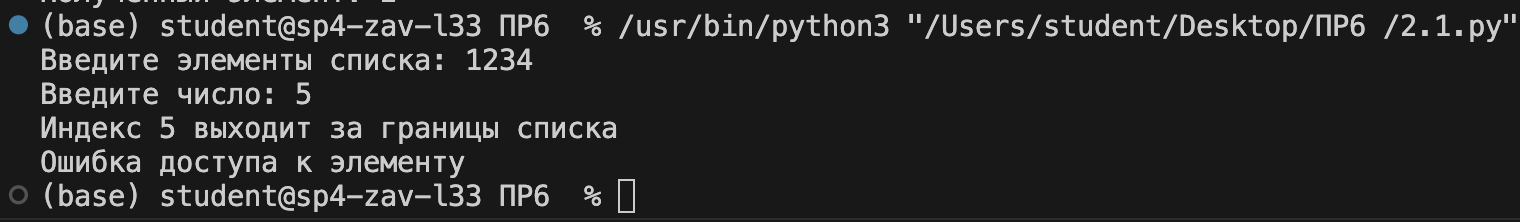
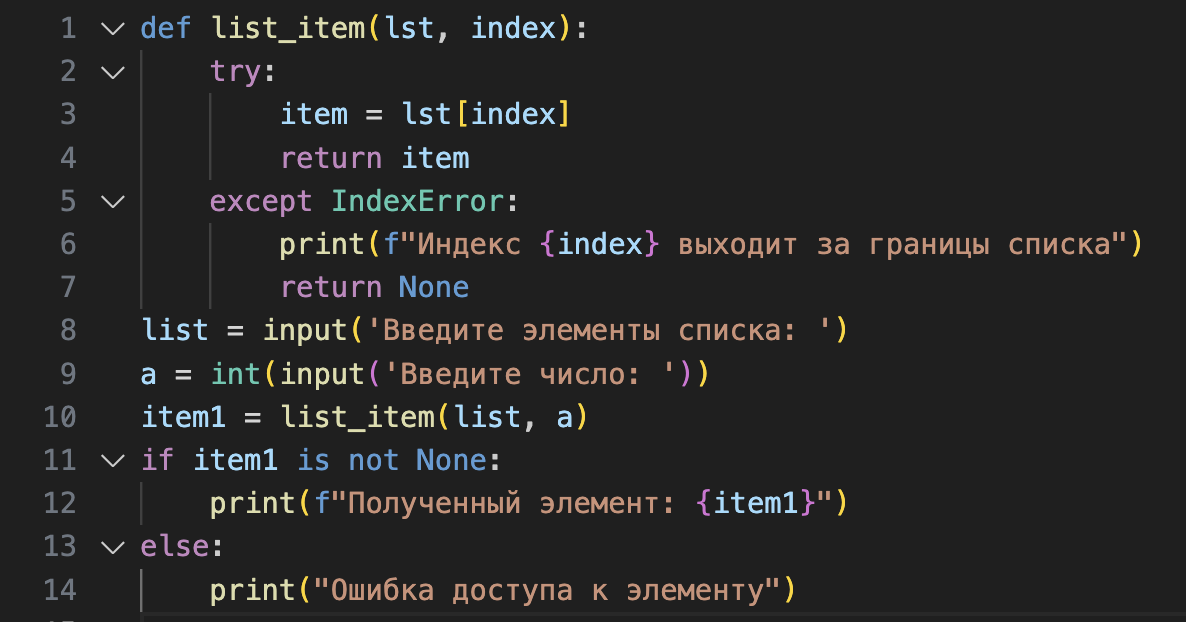
Задание 1: Базовая обработка исключений

1. Создали простой программный модуль, в котором возникает ошибка (например, деление на ноль).

2. Реализовали обработку исключения с помощью конструкции try-catch, чтобы предотвратить аварийное завершение программы

3. Вывели сообщение об ошибке и продолжите выполнение программы.

Задание 2: Иерархия исключений

Создали несколько примеров обработки различных исключений

3. Задокументировали различия между ними и предложили стратегию обработки каждой категории.

Различия между исключениями:

1. Деление на ноль: исключение ZeroDivisionError возникает, когда делитель равен нулю. Обработку стоит выполнять, если есть вероятность передачи нулевого значения в качестве делителя.

2. Выход за границы массива: исключение IndexError возникает, когда запрашивается элемент с индексом, выходящим за границы массива. Важно проверять индексы перед попыткой доступа к элементам массива.

3. Ошибки ввода-вывода: исключения FileNotFoundError, PermissionError и UnicodeDecodeError связаны с операциями над файлами. Нужно учитывать возможные проблемы с наличием файла, правами доступа и правильным декодированием содержимого.

Стратегии обработки исключений:

1. Деление на ноль: проверьте значение делителя перед выполнением операции деления. Если делитель равен нулю, выведите соответствующее сообщение и либо повторите запрос у пользователя, либо завершите операцию.

2. Выход за границы массива: перед обращением к элементу массива убедитесь, что индекс находится в пределах допустимых значений. При выходе за границы массива сообщите об ошибке и предложите корректировку индекса.

3. Ошибки ввода-вывода:

Описание: Исключения, возникающие при работе с файлами, сетевыми соединениями или другими внешними ресурсами. Примеры включают ошибки открытия файла, отсутствие доступа к сети, проблемы с записью данных и т.д.

Обработка:

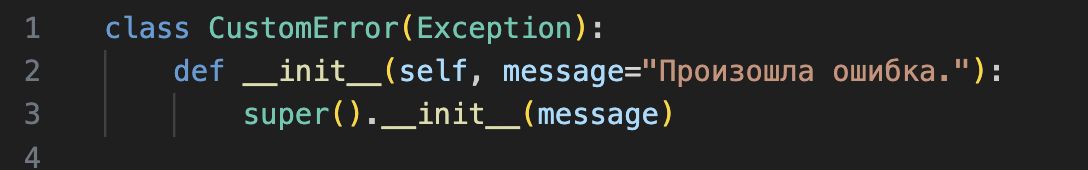
Логгирование всех возникающих ошибок ввода-вывода для последующего анализа.

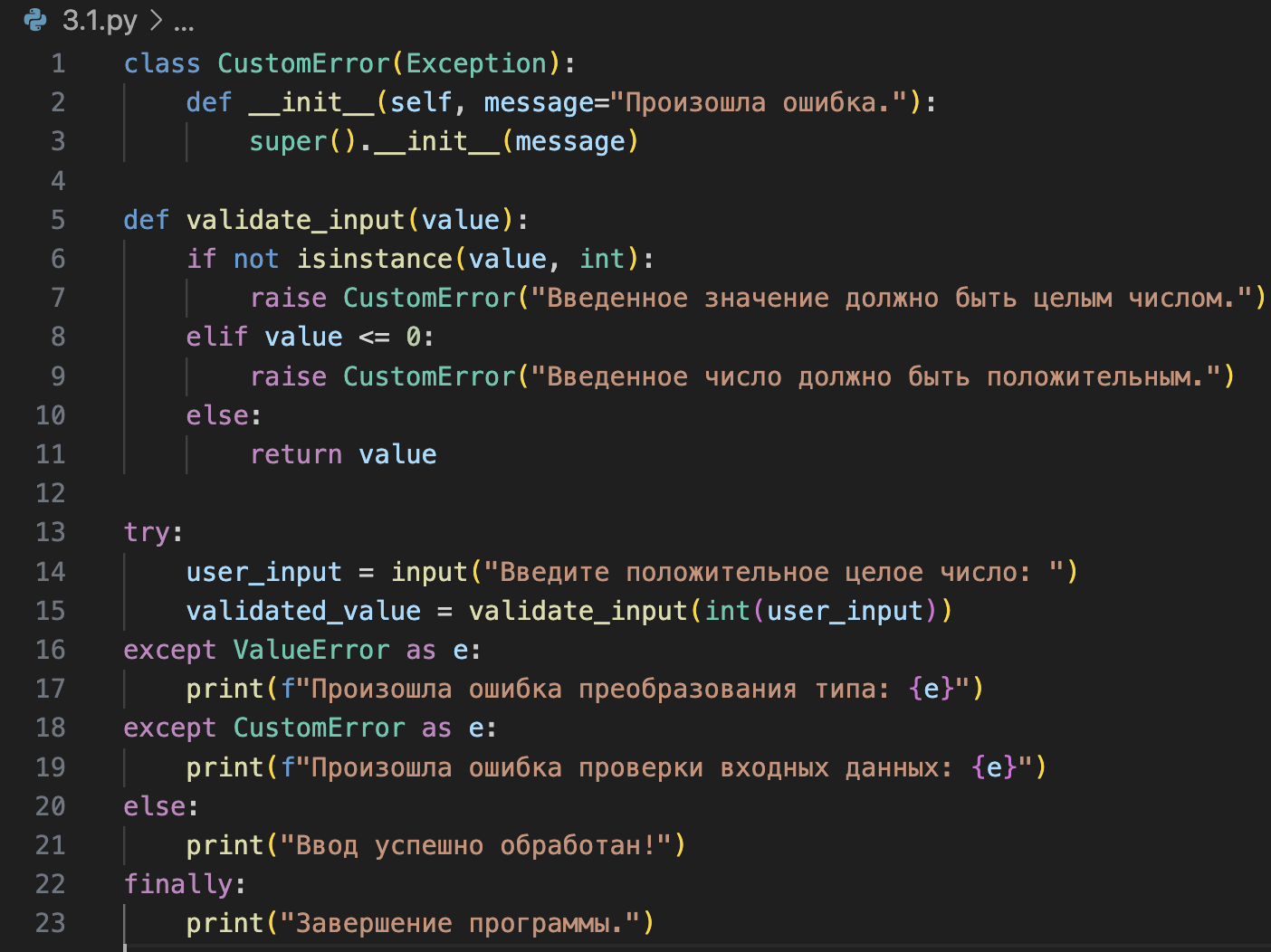
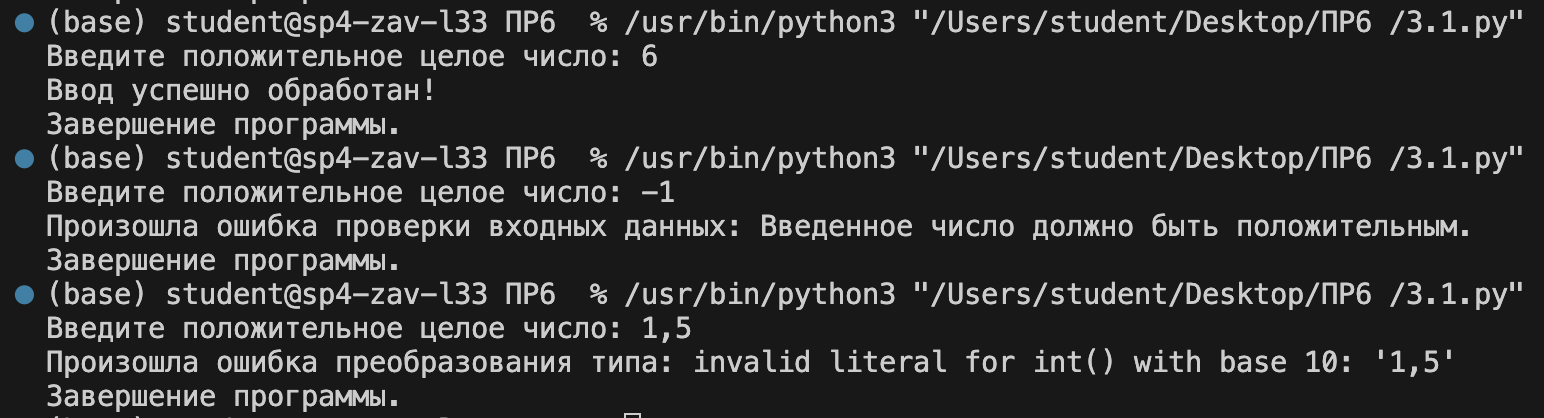
Повторная попытка выполнения операции после возникновения ошибки (retry strategy). Например, при временной недоступности сервера можно подождать несколько секунд и повторить запрос.

Проверка наличия необходимых ресурсов перед выполнением операций. Например, проверка существования файла перед попыткой записи в него.

Обеспечение корректного закрытия файлов и соединений после завершения работы с ними, чтобы избежать утечек ресурсов.

Задание 3: Генерация пользовательских исключений

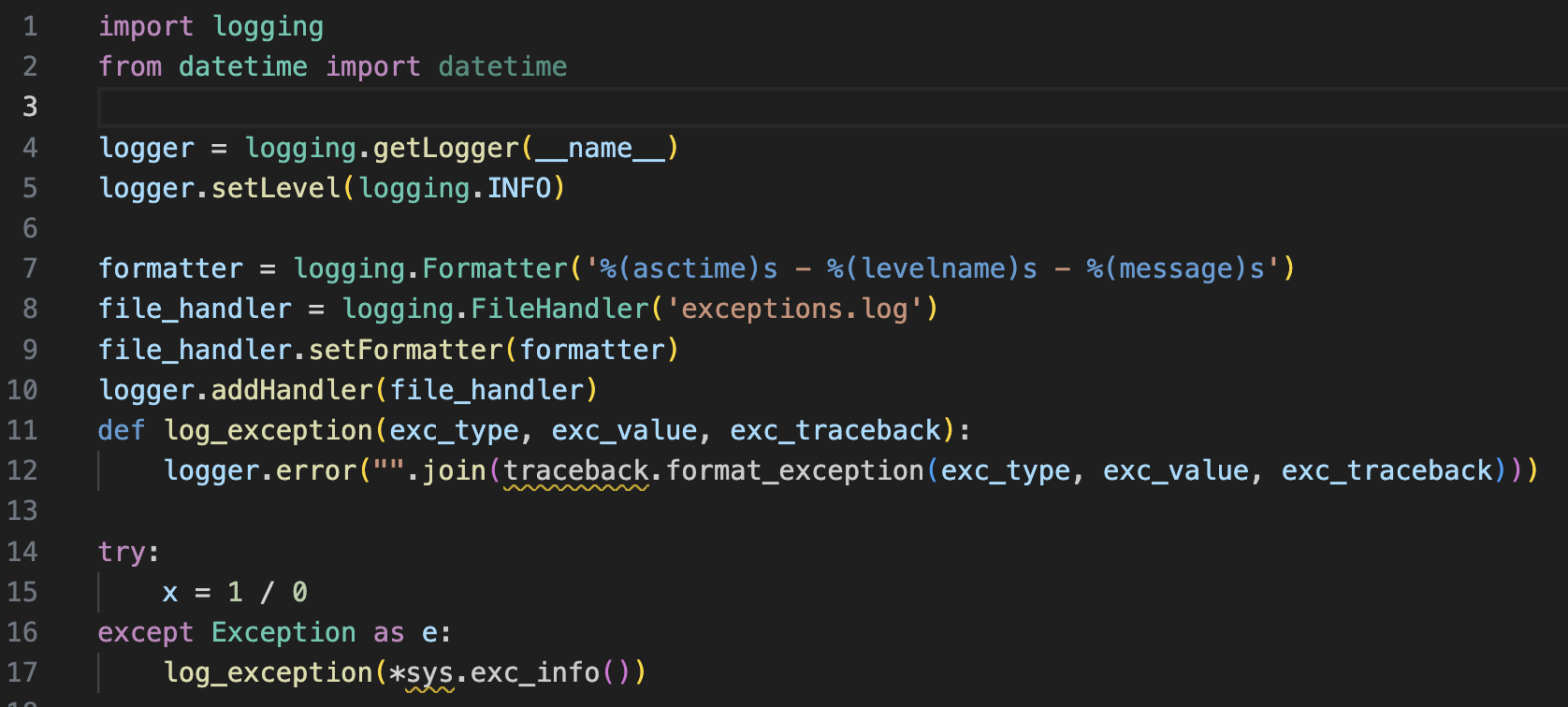
1. Создали пользовательский класс исключений для обработки специфических ситуаций в вашем программном проекте.

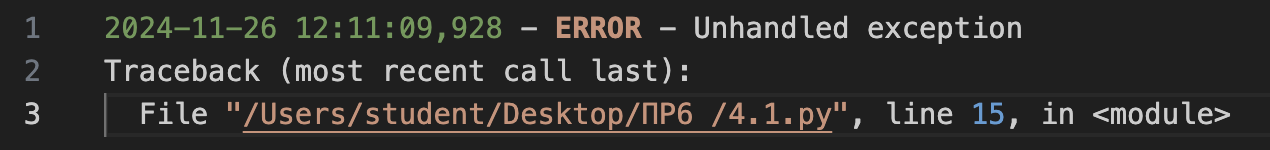
2. Написали пример кода, который генерирует и обрабатывает это исключение.

3. Обосновали необходимость создания пользовательского исключения.

Пользовательские исключения делают код более читаемым и понятным. Разработчики сразу видят, какая конкретная ситуация вызвала ошибку, благодаря чёткому названию и сообщению исключения. Они позволяют точно контролировать, где и как должны обрабатываться определённые виды ошибок и облегчают диагностику проблем, так как они предоставляют более детальную информацию о том, что пошло не так. Также пользовательские исключения помогают создавать модульный код, который легко расширять и изменять. Если в будущем потребуется добавить новые правила проверки или изменить логику обработки ошибок, это можно сделать без значительных изменений в основной части приложения.

Задание 4: Логирование исключений

1. Реализовали механизм логирования исключений в файл или в базу данных для их дальнейшего анализа.

2. Проверили корректность логирования и убедитесь, что каждая ошибка записывается с указанием времени, типа ошибки и контекста.

3. Задокументировали подходы к анализу и устранению исключений на основе логов.

Анализ логов

1. Сортировка логов по типу ошибки поможет быстро найти повторяющиеся проблемы и сосредоточиться на устранении наиболее частых ошибок.

2. Подробное изучение стек-трейсов и контекста ошибок поможет выявить причины их возникновения и места в коде, где они происходят.

3. Анализ временных меток и частоты появления ошибок может помочь обнаружить зависимости между различными событиями и условиями, приводящими к ошибкам.

Устранение исключений

1. На основании анализа логов следует внести изменения в код, чтобы устранить причину возникновения исключений. Это может включать проверку входных данных, добавление дополнительных проверок условий и улучшение обработки ошибок.

2. Внедрение механизмов защиты от возможных ошибок, таких как обработка исключительных ситуаций с помощью блоков try-except, использование assert'ов для проверки предположений и ограничение диапазонов допустимых значений.

3. Анализ логов может также выявить узкие места в производительности, вызванные частыми исключениями.

Документация подходов

1. Описание процесса просмотра и фильтрации логов, включая примеры команд и запросов для поиска нужных данных.

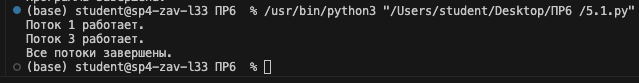
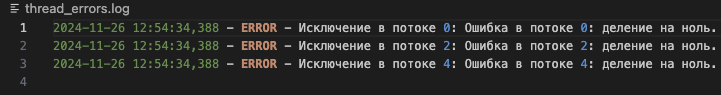
2. Предоставление примеров типичных исправлений для часто встречающихся ошибок, чтобы разработчики могли быстрее находить решения.

3. Описать лучшие практики написания кода, которые помогут минимизировать вероятность возникновения исключений, такие как использование утверждений, валидации данных и обработки ошибок.

Задание 5: Обработка исключений в многопоточности

1. Реализовали обработку исключений в многопоточной среде.



2. Убедились, что исключения в одном потоке не приводят к завершению программы или других потоков. Использовали синхронизацию для корректной работы с исключениями в многопоточных приложениях.

**Выводы работы**

В ходе выполнения практической работы мы изучили методы и подходы к обработке исключений, научились применять конструкции try-catch, обрабатывать и регистрировать исключения для повышения стабильности программного обеспечения.