**Отчет по самостоятельной работе №6**

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

Выполнил: студент

группы 319

Зубарева Екатерина Олеговна

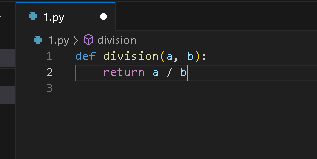
Дата 02.12.2024

# Цель работы: изучить методы и подходы к обработке исключений, научиться применять конструкции try-catch, обрабатывать и регистрировать исключения для повышения стабильности программного обеспечения.

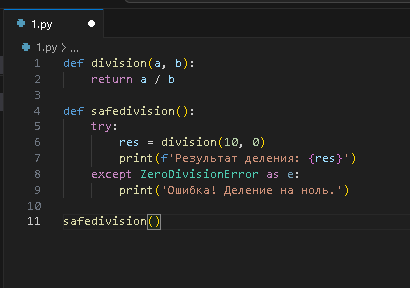
# Основная структура задания:

Задание №1: Базовая обработка исключений

1. Создайте простой программный модуль, в котором возникает ошибка деления на нуль.



1. Реализуйте обработку исключения с помощью конструкции try-catch, чтобы предотвратить аварийное завершение программы. Выведите сообщение об ошибке и продолжите выполнение программы.

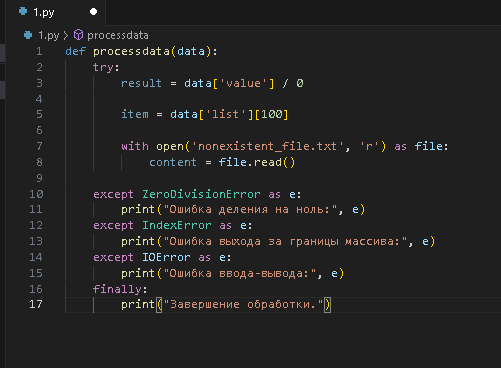


Задание №2: Иерархия исключений

1. Изучите, как работают различные типы исключений в вашем языке программирования (например, стандартные исключения и их наследники).

В Python существует иерархия стандартных исключений:

1. ZeroDivisionError: Ошибка при делении на ноль.
2. IndexError: Выход за пределы допустимого диапазона индексации.
3. IOError: Ошибки ввода/вывода.
4. ValueError: Некорректное значение аргумента функции.
5. Создайте несколько примеров обработки различных исключений (например, деление на ноль, выход за границы массива, ошибки ввода-вывода).



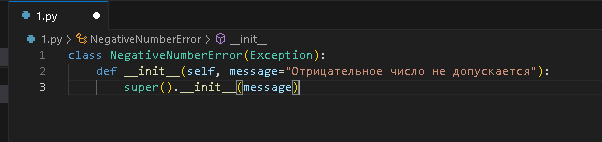
1. Задокументируйте различия между ними и предложите стратегию обработки каждой категории.

Различные исключения возникают в зависимости от типа операции:

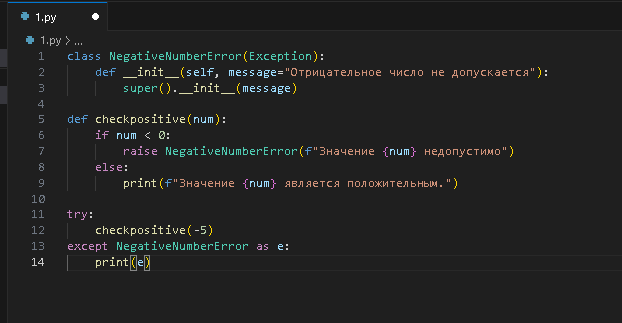
1. Деление на ноль (ZeroDivisionError) — обработчик может либо вернуть специальное значение (например, None), либо предложить альтернативную операцию.
2. Выход за границы массива (IndexError) — можно обработать ошибку, проверяя индекс перед доступом к массиву.
3. Ошибки ввода-вывода (IOError) — такие ошибки могут требовать повторных попыток чтения или записи, либо уведомления пользователя о проблеме.

Задание №3: Генерация пользовательских исключений

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



1. Напишите пример кода, который генерирует и обрабатывает это исключение.

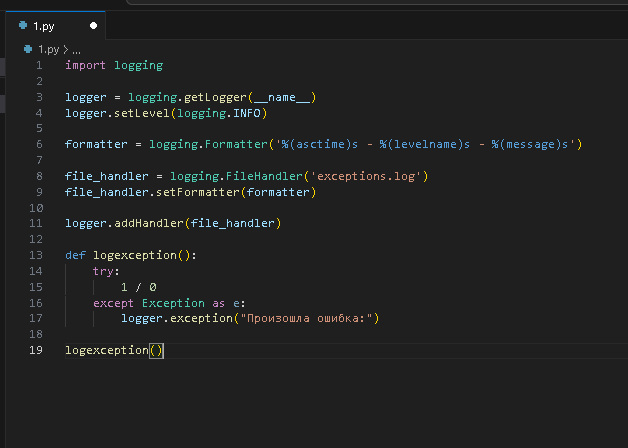


1. Обоснуйте необходимость создания пользовательского исключения.

Пользовательские исключения полезны для того, чтобы отделить специфические ошибки проекта от общих системных ошибок. Это позволяет легче находить и исправлять проблемы, так как контекст ошибки становится яснее.

Задание №4: Логирование исключений

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

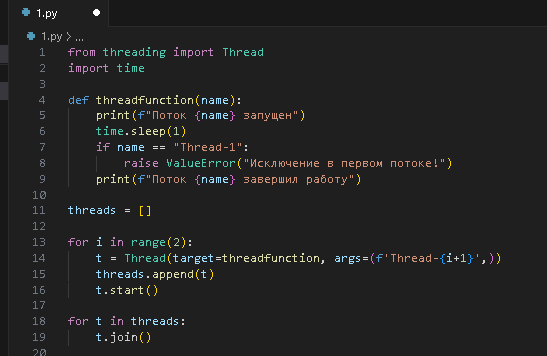


1. Задокументируйте подходы к анализу и устранению исключений на основе логов.

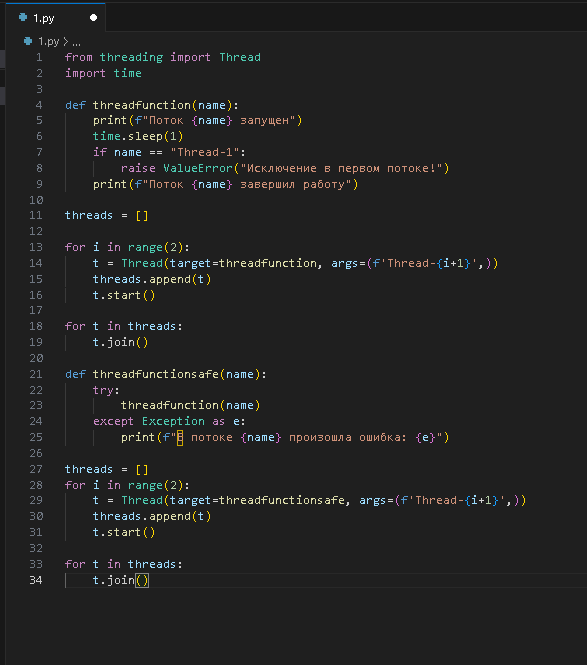
Лог-файлы позволяют отслеживать все возникающие ошибки, включая время возникновения, тип ошибки и стек вызовов. Это помогает выявлять узкие места в программе и устранять баги.

Задание №5: Обработка исключений в многопоточности

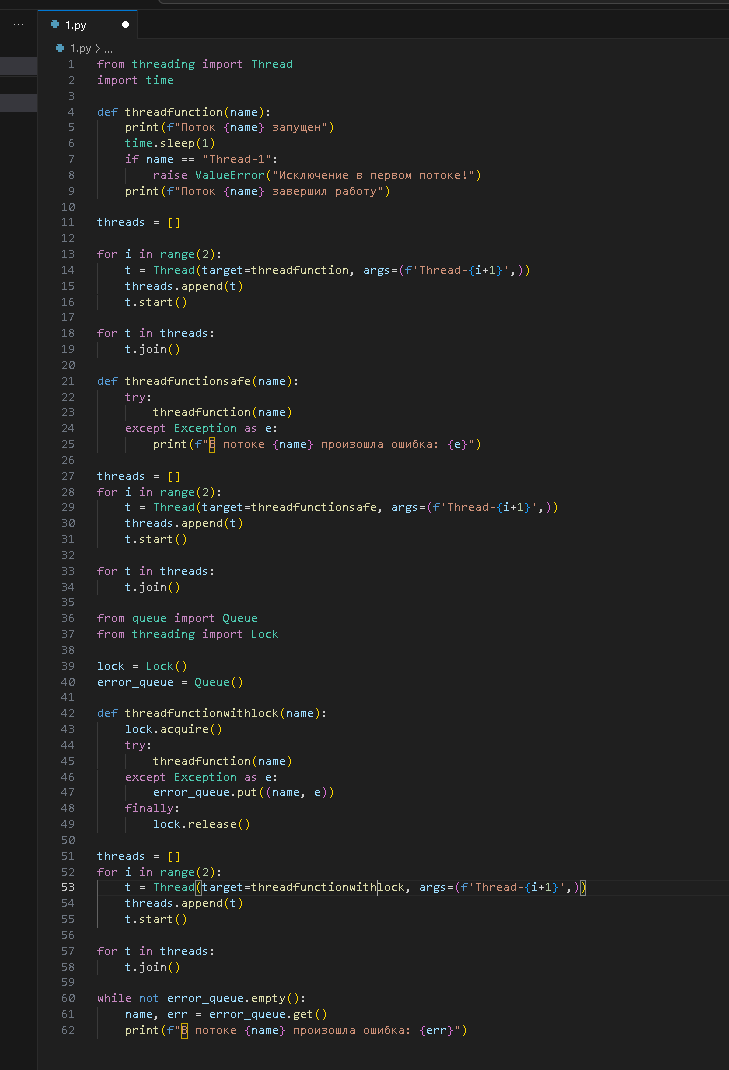
1. Если ваш проект использует многопоточность, реализуйте обработку исключений в многопоточной среде.



1. Убедитесь, что исключения в одном потоке не приводят к завершению программы или других потоков.



1. Используйте синхронизацию для корректной работы с исключениями в многопоточных приложениях.



# Выводы работы:

# В ходе работы, я изучила методы и подходы к обработке исключений, научилась применять конструкции try-catch, обрабатывать и регистрировать исключения для повышения стабильности программного обеспечения.