**Отчет по самостоятельной работе №7**

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

Выполнил: студент

группы 319

Меркушин Евгений Романович

Дата 17.11.2024

**Цель работы**

Цель данной работы – изучить и применить отладочные классы в проекте для улучшения процесса поиска и исправления ошибок в коде.

**Основная структура задания**

# Задание 1: Создание отладочных классов

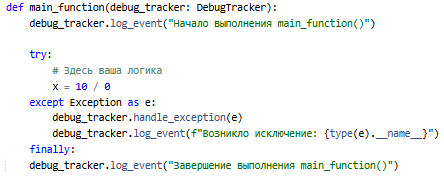
1. **Класс для отслеживания выполнения программы**: Начнем с простого класса, который будет использоваться для отслеживания состояния программы и логирования событий.

****

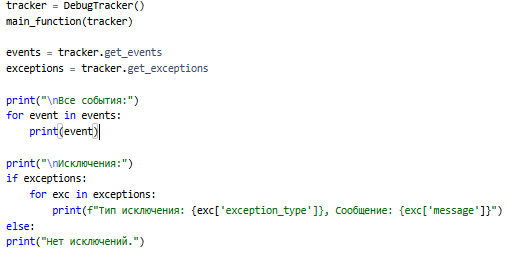
**Добавление логирования исключений**: Класс уже содержит метод handle\_exception, который регистрирует исключения вместе с информацией о типе исключения, сообщении и трассировке.

# Задание 2: Отладка проекта с использованием отладочного класса

1. **Внедрение отладочного класса в основной проект**:Предположим, у нас есть основная функция, которая выполняет какую-то операцию и может генерировать исключения.

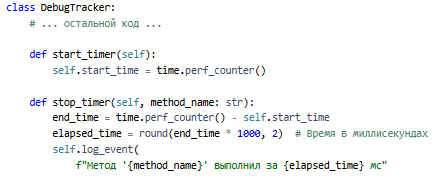
****

**Тестирование с использованием функционала класса**:Теперь создадим экземпляр класса DebugTracker и вызовем основную функцию.

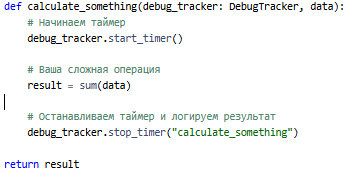


# Задание 3: Использование отладочного класса для анализа производительности

1. **Возможность измерения времени выполнения методов**:Расширим наш класс DebugTracker, добавив возможность измерять время выполнения методов.

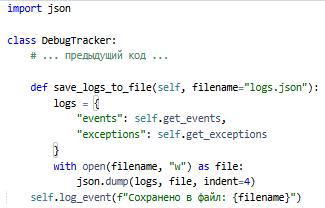


1. **Анализ производительности одного из модулей проекта**:Допустим, мы хотим проанализировать производительность метода calculate\_something() в нашем проекте.



# Задание 4: Интеграция отладочного класса с логированием

1. **Добавление возможности автоматического сохранения логов в файл**:Добавим новый метод в класс DebugTracker.

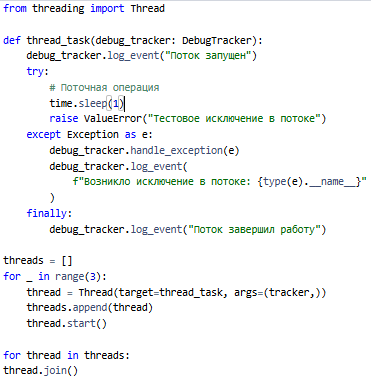


**Проверка корректности логирования в проекте**:После выполнения основных функций вызываем метод save\_logs\_to\_file.

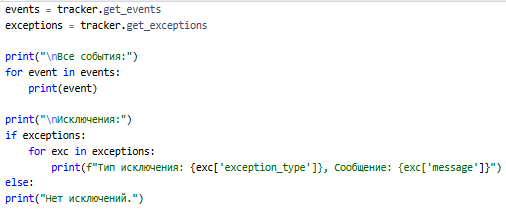


# Задание 5: Отладка многопоточной программы с помощью отладочного класса

1. **Применение отладочного класса для отслеживания ошибок в многопоточной программе**:Используем класс Thread для создания нескольких потоков и передачи экземпляра DebugTracker каждому потоку.



**Контроль корректности отслеживания ошибок в разных потоках**:Посмотрим на логи, сохранённые в объекте DebugTracker, чтобы убедиться, что все исключения и события зарегистрированы.



**Вывод**

Мы создали универсальный отладочный класс DebugTracker, который способен отслеживать выполнение программы, регистрировать исключения, измерять производительность и сохранять логи в файл. Кроме того, продемонстрировали, как этот класс можно использовать в многопоточных программах для контроля ошибок и событий.

Начало формы