Что сдавать?

Архив, содержащий:

- исправленный с`calc.py`;
- написанный `fuzz target` для метода `calculate`;
- файлы с содержимым консоли после завершения работы инструмента до исправления очередной ошибки (на каждую исправленную ошибку);
- документ с отчетом, содержащим описания ошибок и способы их исправления.

Задание

При помощи фаззинга проанализировать калькулятор `calc.py` с использованием инструмента **`hypothesis`**. Для этого нужно написать цель для фаззинга (`fuzz target`) для метода `calculate`. Случайная строка при этом должна быть предварительно обработана методом `opn`. Все обнаруженные ошибки необходимо описать и исправить. Ошибочными считаются все сценарии, когда калькулятор выбрасывает исключение, отличное от `CalcException\OverflowError\ValueError`. То есть `CalcException и другие` — предусмотренные при разработке(и исходящие из ограничений подхода при разработке калькуятора) ошибочные сценарии, все остальное — ошибки.

Команда запуска(для файла fuzz с соответствующим fuzz target):

```
pytest --hypothesis-show-statistics .\fuzz.py
```

Задание на 10

Для получения оценки >= 8 необходимо также сделать следующее:

- выбрать open-source библиотеку на Python, реализующую взаимодействие с каким-нибудь популярным форматом данных (xml, yaml, jpeg, raw, mp3, что угодно), добавить соответствующий import и указать версию в requirements.txt;
- написать для нее 3 `fuzz targets` (для разных методов), добавить соответствующий алфавит;
- запустить фаззинг (желательно, хотя бы на несколько часов);
- сдать проект с `fuzz targets`.

Фаззинг для Python чуть менее распространен на практике, чем для C/C++ проектов, поэтому шанс найти какую-нибудь настоящую ошибку вполне существует!