Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «ИНЖЕНЕРНЫЙ БИЗНЕС И МЕНЕДЖМЕНТ»

КАФЕДРА «ПРОМЫШЛЕННАЯ ЛОГИСТИКА» (ИБМ-3)

Рубежный Контроль №2

«Парадигмы и конструкции языков программирования»

38.03.05 «Бизнес-информатика» (бакалавриат)

Студент ИБМ3-34Б _____ Сергеева М. А.
(Подпись, дата)

2024 г.

1. Введение

В рамках рубежного контроля по дисциплине программирования была поставлена задача: **рефакторинг исходной программы для модульного тестирования** и **создание модульных тестов с использованием TDD-фреймворка**. В ходе работы была произведена оптимизация исходного кода, а также написаны тесты для проверки корректности выполнения основных функций программы.

2. Цель работы

Целью работы было:

- **Рефакторинг программы** для обеспечения удобства модульного тестирования.
- **Создание модульных тестов** с использованием фреймворка `unittest` в стиле TDD (Test-Driven Development).

3. Описание исходной программы

Исходная программа, предоставленная для выполнения задания, состоит из трех классов:

- **SyntaxConstruction** — описывает синтаксические конструкции для языков программирования.
- **ProgrammingLanguage** — описывает языки программирования.
- **LanguageSyntax** — связывает синтаксические конструкции с языками программирования.

Программа выполняет следующие операции:

1. Выводит список синтаксических конструкций для каждого языка, отсортированный по языкам.
2. Выводит список языков с количеством синтаксических конструкций.
3. Фильтрует языки, содержащие слово "язык" в названии, и выводит их синтаксические конструкции.

4. Шаги выполнения работы

4.1. Рефакторинг программы

Для того чтобы программа стала пригодной для модульного тестирования, был произведен рефакторинг исходного кода. Вместо того чтобы выполнять все действия непосредственно в основной части программы, была произведена следующая реорганизация:

1. Разделение логики на отдельные функции:
    - `get_syntax_constructions_by_language`: возвращает список синтаксических конструкций для каждого языка.
    - `get_language_syntax_count`: подсчитывает количество синтаксических конструкций для каждого языка.
    - `get_filtered_languages`: фильтрует языки, содержащие слово "язык" в названии, и выводит их синтаксические конструкции.

4.2. Разработка тестов

После рефакторинга была написана серия модульных тестов с использованием фреймворка `unittest`. Тесты включают:

- **Тест 1**: Проверка функции `get_syntax_constructions_by_language`. Тестирует правильность группировки синтаксических конструкций по языкам.
- **Тест 2**: Проверка функции `get_language_syntax_count`. Тестирует правильность подсчета количества синтаксических конструкций для каждого языка.
- **Тест 3**: Проверка функции `get_filtered_languages`. Тестирует фильтрацию языков, содержащих слово "язык" в названии, и правильность вывода связанных синтаксических конструкций.

5. Результаты выполнения работы

Все тесты были успешно выполнены, что подтверждает корректность работы программы. Результаты выполнения тестов:

```
Ran 3 tests in 0.000s

OK
```

Программы:

1. **Основной код** (файл `syntax_program.py`): Рефакторинг программы для обеспечения удобства тестирования.
2. **Тесты** (файл `test_syntax_program.py`): Создание и запуск модульных тестов с использованием `unittest`.

6. Заключение

В результате выполнения работы была проведена реорганизация исходной программы для модульного тестирования, а также созданы тесты для проверки ключевых функций программы. Все тесты успешно прошли, что подтверждает корректность работы программы.