# Тестовый план для парсинга сайтов.

1. Тестирование функциональности парсинга:

- Проверка возможности программы получить доступ к новостному сайту.

- Проверка правильности извлечения топиков с новостного сайта.

2. Тестирование качества данных:

- Проверка точности извлечения информации о топиках.

- Проверка полноты данных, полученных в результате парсинга.

- Проверка соответствия данных формату, указанному в требованиях.

3. Тестирование производительности:

- Проверка времени, затраченного на парсинг одного топика.

- Проверка скорости работы программы при парсинге большого объема данных.

# Тестирование функциональности парсинга.

1. Получение html-кода страницы: get\_html()  
   Функция принимает url в качестве строки и возвращает html-содержимое страницы.  
   Реализованные тест-кейсты -- проверка различных хостов:
   1. 'https://journal.tinkoff.ru/'
   2. None
   3. 'https://ya.ru'
   4. 'htps://ya.ru'
   5. numeric\_host = 5051

Изначально функция некорректно работала на тестах b, d, e.

1. Получение flows (топиков) со спаршенной страницы. Функция принимает html-код страницы и парсит топики оттуда по заданному шаблону.   
     
   Проверка различных спаршенных html показала, что функция получения топиков работает корректно в случаях, когда передается HTML неожиданного формата. Если топики не найдены, то вернется просто пустой DataFrame. Поэтому проверки на случайный html, пустой html были пройдены.

Тест-кейсы:

* 1. HTML для https://journal.tinkoff.ru/
  2. NULL
  3. Пустой HTML
  4. HTML для htps://ya.ru

Единственный тест-кейс когда функция не работала корректно – Null вместо html.

1. Функциональность парсинга топиков. Функция принимает спаршенные топики и выделяет соответствующие им сущности.

Тест-кейсы:

* 1. Топики для 'https://journal.tinkoff.ru/
  2. NULL топики
  3. Пустой массив топиков
  4. Случайные топики

Все тест-кейсы были реализованы в виде Юнит-Тестов с помощью библиотеки unittests для Python3 и отправлены разработчику.

# Тестирование качества данных.

В силу специфики разработанного парсера авто-тесты на соответствие данных разрабатывать достаточно проблематично. Поэтому для проверки соответствия данных было проведено ручное тестирование.

Данные в полученном датафрейме соответствуют тому, что действительно есть на сайте.

Единственная проблема, которая была выявлена при проведении ручного тестирования данных – html со временем меняет теги классов, видимо владельцы сайта специально так делают, чтобы их данные не воровали. Проблема была сообщена разработчику, после чего, было принято решение сделать временную правку в виде конфига с фиксированными названиями классов, который можно быстро поменять. На дальнейших этапах разработки планируется устранять эту проблему автоматически.

# Тестирование производительности

Тестирование производительности производилось на сервере со следующими характеристиками:

ubuntu 22.04  
RAM: 16GB

cores: 4cpu

Прохождение всех тест-кейсов: 162s

Работа пайпланай: 50s

# Тестовый пайплайна парсинга документов.

Для тестирования парсинга документов будет описано ручное тестирование результатов, так как тестирование функциональности проводилось как в предыдущем парсере.

Тест кейсы:

1. Пустой файл
2. Файл со сломанной кодировкой
3. Файл с обычным текстом
4. Битый файл

Были поданы файлы форматов .docx, .doc, .pdf