**САНКТ-ПЕТЕРБУРГСКИЙ КОЛЛЕДЖ ИНФОРМАЦИОНЫХ ТЕХНОЛОГИЙ**

**Отчет по практическому заданию**

**МДК 05.03 Тестирование информационных систем**

Выполнил

студент группы 415:

Пирогов В.А.

Проверила Полякова А. Н.

Оценка \_\_\_\_\_\_\_\_

Санкт-Петербург 2025

**Цель:** получить навыки определения вида и ситуации тестирования.

**Задание 1**

Определить, к какому виду тестирования относится ситуация. Заметьте, что тестируемая ситуация относится к нескольким видам тестирования.

|  |  |  |
| --- | --- | --- |
| N | Тестируемая ситуация | Вид тестирования |
| 1 | Тестировщиков пригласили в проект и дали для ознакомления различную документацию, связанную с проектом | Статистическое, дымовое, тестирование критического пути, позитивное тестирование. |
| 2 | Регистрация пользователя на сайте | Позитивное тестирование, функциональное тестирование |
| 3 | Проверяем, что при бездействии пользователя на сайте его сессия истекает через 30 минут, выставив статус «Close» в БД | Функциональное тестирование, ручное тестирование, метод серого ящика |
| 4 | Встретились с предполагаемым пользователем системы, дали ему руководство на выполнение частых операций и наблюдали, как он их выполняет. Записали все замечания и неудобства, которые описывал пользователь. | Юзабилити-тестирование, метод черного ящика, функциональное тестирование |
| 5 | Скорость отклика приложения, когда сайт используют 100 пользователей, что на 10% больше, чем заявлено в требованиях. | Тестирование стабильности и надежности, нагрузочное тестирование |
| 6 | Скорость отклика и возможность производить на сайте операции, когда сайт используют 10000 пользователей, что в 4 раза больше ежедневной предполагаемой нагрузки. | Объемное тестирование, стресс-тестирование |
| 7 | Перенесли тестируемое приложение в окружение, подобное тому, что у заказчика, и увеличили объем БД, с которой работает БД, в 10 раз. | Объемное тестирование, интеграционное тестирование, регрессионное тестирование |
| 8 | Тестирование перед тем, как отдать версию заказчику, с проверкой того, что ранее разработанный функционал работает и выполняет операции так же, как и раньше. | Дымовое тестирование, пути, расширенное тестирование, негативное тестирование |
| 9 | Получили новую версию ПО, необходимо быстро ответить, годится ли она для дальнейшего тестирования. | Дымовое тестирование, метод серого ящика, регрессионное тестирование |
| 10 | Загрузка файла с расширением «.txt» на YouTube | Функциональное тестирование, негативное тестирование |

* **Задание 2**

Выбрать тестируемое приложение и придумать по три теста следующих видов. Фантазию включать можно и нужно, без нее в тестирование никак.

**Тестируемое приложение:** Яндекс почта

|  |  |  |
| --- | --- | --- |
| N | Вид тестирование | Тестируемая ситуация |
| 1 | Функциональное тестирование методом черного ящика | 1.Тестированние отправки и получение писем  2. Тестирование функционала фильтрации писем  3. Тестирование взаимодействия с вложениями |
| 2 | Юзабилити-тестирование | 1. Тестирование удобства навигации  2. Тестирование удобства процесса создания и форматирование писем  3. Тестирование реакции интерфейса на ошибки пользователя |
| 3 | Негативное функциональное тестирование | 1. Ввод неправильного формата адреса электронной почты  2. Неправильное заполнение полей при регистрации  3. Неправильный ввод пароля для входа |
| 4 | Нагрузочное тестирование | 1. Одновременная отправка большого количества электронных писем  2. Открытие писем с большими вложениями  3. Отправка большого вложения |
| 5 | Дымовое тестирование | 1. Запуск приложения после обновления  2. Проверка основных функций приложения  3. Проверка взаимодействия приложения с другими компонентами или сервисами |
| 6 | Конфигурационное тестирование | 1. Тестирование работы при минимальных требованиях приложения к характеристикам компьютера  2. Тестирование приложения при работе с другой операционной системы  3. Тестирование работы приложения при слабом интернет-соединении |
| 7 | Тестирование критического пути | 1. Отправка большого количества сообщений с вложениями одновременно  2. Массовое получение писем с одновременным открытием вложений  3. Синхронизация сообщений и контактов на различных устройствах |
| 8 | Инсталляционное тестирование | 1. Установка приложения на различные операционные системы  2. Обновление приложения до последней версии  3. Деинсталляция приложения |

**Практическая работа номер 5**

Задание 1

**Системные характеристики**

• СХ-1: Приложение является консольным.

• СХ-2: Для работы приложение использует интерпретатор PHP.

Каким образом приложение использует интерпретатор PHP?

Приложение использует интерпретатор PHP для выполнения серверной части кода. Интерпретатор PHP выполняет PHP-скрипты, интерпретируя их и обрабатывая запросы от клиентской стороны.

• СХ-3: Приложение является кроссплатформенным.

Какие именно платформы должно поддерживать приложение?

Платформы, которые должно поддерживать приложение: Windows и Linux.

**Пользовательские требования**

• Также см. диаграмму вариантов использования.

• ПТ-1: Запуск и остановка приложения.

* ПТ-1.1: Запуск приложения производится из консоли командой «PHP

converter.php параметры».

* ПТ-1.2: Остановка приложения производится выполнением команды

Ctrl+C.

• ПТ-2: Конфигурирование приложения.

* ПТ-2.1: Конфигурирование приложения сводится к указанию путей в

файловой системе.

Какие именно пути необходимо указать в файловой системе?

Пути, которые необходимо указать в файловой системе, определяются требованиями приложения и его функциональностью

* ПТ-2.2: Целевой кодировкой является UTF8.

ПТ-3: Просмотр журнала работы приложения.

* ПТ-3.1: В процессе работы приложение должно выводить журнал

своей работы в консоль и лог-файл.

Какие именно события и данные должны выводится?

Это могут быть различные уведомления, сообщения об ошибках, отчеты о выполнении задач или другая информация, которую приложение должно предоставлять пользователю

* ПТ-3.2: При первом запуске приложения лог-файл создаётся, а при

последующих — дописывается.

**Бизнес-правила**

• БП-1: Источник и приёмник файлов

* БП-1.1: Каталоги, являющиеся источником исходных и приёмников

конечных файлов не должны совпадать.

Как определяются исходные и конечные файлы?

Исходные и конечные файлы определяются функциональностью приложения. Исходные файлы обычно содержат исходный код приложения, написанный на языке программирования PHP

* БП-1.2: Каталог, являющийся приёмником конечных файлов, не

может быть подкаталогом источника.

**Атрибуты качества**

• АК-1: Производительность

* АК-1.1: Приложение должно обеспечивать скорость обработки

данных 5 МБ/сек.

Каким образом определяется скорость обработки данных?

Скоростью ответа приложения на запрос пользователя

• АК-2: Устойчивость к входным данным

* АК-2.1: Приложение должно обрабатывать входные файлы размером

до 50 МБ включительно.

* АК-2.2: Если входной файл не является текстовым, приложение

должно произвести обработку.

Каким образом приложение выполняет обработку файлов, которых не получается преобразовать в текстовый форматов файлов?

Приложение пропускает данные файлы, выводя ошибку преобразования

Задание 2

**Системные характеристики**

• СХ-1: Приложение является консольным.

• СХ-2: Приложение разрабатывается на языке программирования PHP

(причина выбора языка PHP отражена в пункте О-1 раздела

«Ограничения», особенности и важные настройки интерпретатора PHP

отражены в пункте ДС-1 раздела «Детальные спецификации»).

• СХ-3: Приложение является кроссплатформенным с учётом пункта О-4

раздела «Ограничения».

**Пользовательские требования**

• Также см. диаграмму вариантов использования.

• ПТ-1: Запуск и остановка приложения.

o ПТ-1.1: Запуск приложения производится из консоли командой «php

converter.php SOURCE\_DIR DESTINATION\_DIR [LOG\_FILE\_NAME]»

(описание параметров приведено в разделе ДС-2.1, реакция на

ошибки при указании параметров приведена в разделах ДС-2.2, ДС-

2.3, ДС-2.4).

o ПТ-1.2: Остановка приложения производится выполнением команды

Ctrl+C в окне консоли, из которого было запущено приложение.

• ПТ-2: Конфигурирование приложения.

o o ПТ-2.1: Конфигурирование приложения сводится к указанию

параметров командной строки (см. ДС-2).

o o ПТ-2.2: Целевой кодировкой преобразования текстов является

кодировка UTF8 (также см. О-5).

Какая кодировка является исходной для преобразования?

Кодировка, которая стоит по умолчанию.

• ПТ-3: Просмотр журнала работы приложения.

o o ПТ-3.1: В процессе работы приложение должно выводить журнал

своей работы в консоль и лог-файл (см. ДС-4), имя которого

определяется правилами, указанными в ДС-2.1.

o ПТ-3.2: Формат журнала работы и лог файла указан в ДС-4.1, а реакция

приложения на наличие или отсутствие лог-файла указана в ДС-4.2 и

ДС-4.3 соответственно.

**Бизнес-правила**

• БП-1: Источник и приёмник файлов

o БП-1.1: Каталоги, являющиеся источником исходных и приёмником

конечных файлов, не должны совпадать (см. также ДС-2.1 и ДС-3.2).

o БП-1.2: Каталог, являющийся приёмником конечных файлов, не

может находиться внутри каталога, являющегося источником

исходных файлов или его подкаталогов (см. также ДС-2.1 и ДС-3.2).

**Атрибуты качества**

• АК-1: Производительность

o АК-1.1: Приложение должно обеспечивать скорость обработки

данных не менее 5 МБ/сек на аппаратном обеспечении,

эквивалентном следующему: процессор i7, 4 ГБ оперативной памяти,

средняя скорость чтения/записи на диск 30 МБ/сек. Также см. О-6.

Какие конкретные операции или функции приложения требуют высокой производительности?

Операции преобразования файла

• АК-2: Устойчивость к входным данным

o АК-2.1: Требования относительно форматов обрабатываемых файлов

изложены в ДС-5.1.

o АК-2.2: Требования относительно размеров обрабатываемых файлов

изложены в ДС-5.2.

o АК-2.3: Поведение приложения в ситуации обработки файлов с

нарушениями формата определено в ДС-5.3.

**Ограничения**

• О-1: Приложение разрабатывается на языке программирования PHP,

использование которого обусловлено возможностью заказчика

осуществлять поддержку приложения силами собственного IT-отдела.

• О-2: Ограничения относительно версии и настроек интерпретатора PHP

отражены в пункте ДС-1 раздела «Детальные спецификации».

• О-3: Процедуры установки и настройки интерпретатора PHP выходят за

рамки данного проекта и не описываются в документации.

• О-4: Кроссплатформенные возможности приложения сводятся к

способности работать под ОС семейства Windows и Linux,

поддерживающих работу интерпретатора PHP версии, указанной в ДС-1.1.

• О-5: Целевая кодировка UTF8 является жёстко заданной, и её изменение

в процессе эксплуатации приложения не предусмотрено.

• О-6: Допускается невыполнение АК-1.1 в случае, если невозможность

обеспечить заявленную производительность обусловлена

объективными внешними причинами (например, техническими

проблемами на сервере заказчика).

Созданные на основе таких пользовательских требований детальные

спецификации имеют следующий вид.

**Детальные спецификации**

**ДС-1: Интерпретатор PHP**

ДС-1.1: Минимальная версия — 5.5.

Как должно вести себя приложение если версия меньше или больше заявленной?

Если версия меньше заявленной, то приложение выводит ошибку несовместимости. Если версия больше, то приложение работает в штатном режиме

ДС-1.2: Для работы приложения должно быть установлено и включено

расширение mbstring.

Как должно вести себя приложение если расширение не установлено или выключено?

Выводить ошибку с указанием на отсутствующее расширение

**ДС-2: Параметры командной строки**

ДС-2.1: При запуске приложения оно получает из командной строки три

параметра:

SOURCE\_DIR — обязательный параметр, определяет путь к

каталогу с файлами, которые необходимо обработать;

DESTINATION\_DIR — обязательный параметр, определяет путь к

каталогу, в который необходимо поместить обработанные файлы

(этот каталог не может находиться внутри каталога SOURCE\_DIR

или в его подкаталогах (см. БП-1.1 и БП-1.2));

LOG\_FILE\_NAME — необязательный параметр, определяет полное

имя лог-файла (по умолчанию лог-файл с именем «converter.log»

размещается по тому же пути, по которому находится файл скрипта

converter.php);

ДС-2.2: При указании недостаточного количества параметров командной

строки приложение должно завершить работу, выдав сообщение об

использовании (ДС-3.1).

ДС-2.3: При указании излишнего количества параметров командной

строки приложение должно игнорировать все параметры командной строки,

кроме указан-ных в пункте ДС-2.1.

ДС-2.4: При указании неверного значения любого из параметров

командной строки приложение должно завершить работу, выдав сообщение

об использовании (ДС-3.1), а также сообщив имя неверно указанного

параметра, его значение и суть ошибки (см. ДС-3.2).

**ДС-3: Сообщения**

ДС-3.1: Сообщение об использовании: «USAGE converter.php SOURCE\_DIR

DESTINATION\_DIR LOG\_FILE\_NAME».

ДС-3.2: Сообщения об ошибках:

Directory not exists or inaccessible.

Destination dir may not reside within source dir tree.

Wrong file name or inaccessible path.

**ДС-4: Журнал работы**

ДС-4.1: Формат журнала работы одинаков для отображения в консоли и

записи в лог-файл: YYYY-MM-DD HH:II:SS имя\_операции параметры\_операции

результат\_операции.

ДС-4.2: В случае если лог-файл отсутствует, должен быть создан новый

пустой лог-файл.

Должно ли приложение проверять наличие лог-файла перед каждой записью, или создание нового пустого лог-файла должно происходить только при его отсутствии в начале работы приложения?

Создание нового пустого лог-файла должно происходить только при его отсутствии в начале работы приложения

ДС-4.3: В случае если лог-файл уже существует, должно происходить

добавление новых записей в его конец.

Следует ли производить какую-либо дополнительную обработку или проверку существующего лог-файла до добавления новых записей, или просто открыть его для записи и добавить новые записи в конец?

Никакая дополнительная обработка не требуется. Необходимо, чтобы приложение просто открывало лог-файл для записи и добавляло новые записи в конец

**ДС-5: Форматы и размеры файлов**

ДС-5.1: Приложение должно обрабатывать текстовые файлы на русском

и английском языках в следующих исходных кодировках: WIN1251, CP866,

KOI8R.

Как должно реагировать приложение на другие кодировки, просто пропускать или выводить ошибку?

Просто пропускать

Обрабатываемые файлы могут быть представлены в следующих форматах,

определяемых расширениями файлов:

Plain Text (TXT);

Hyper Text Markup Language Document (HTML);

Mark Down Document (MD).

Допускается ли приложению обрабатывать только файлы с указанными расширениями, или также разрешено обрабатывать файлы с другими расширениями, но соответствующие указанным форматам?

Допускается обрабатывать файлы с другими расширениями, но с указанными форматами

ДС-5.2: Приложение должно обрабатывать файлы размером до 50 МБ

(включительно), игнорируя любой файл, размер которого превышает 50 МБ.

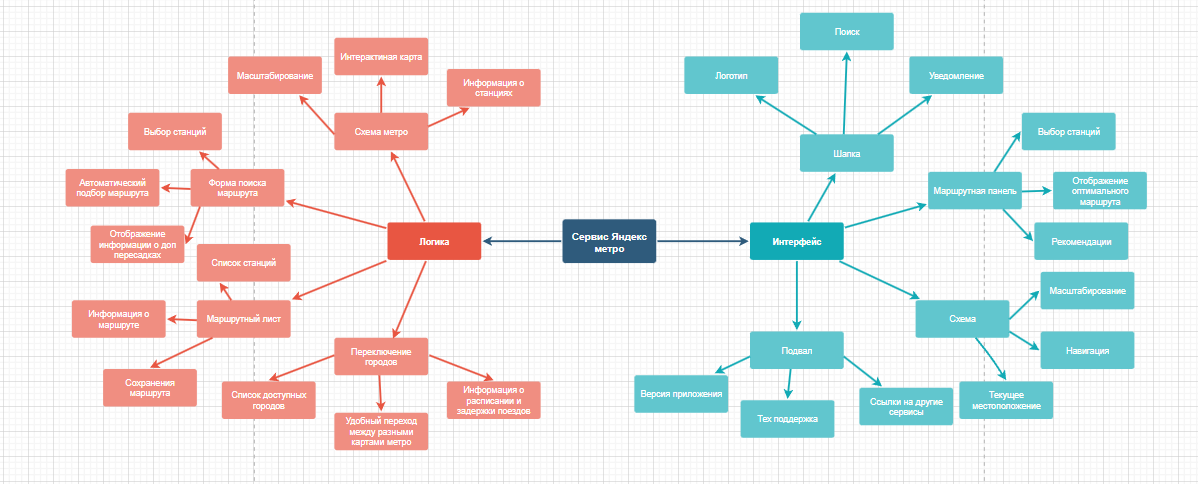
ДС-5.3: Если файл с расширением из ДС-5.1 содержит внутри себя

данные, не соответствующие формату файла, допускается повреждение таких

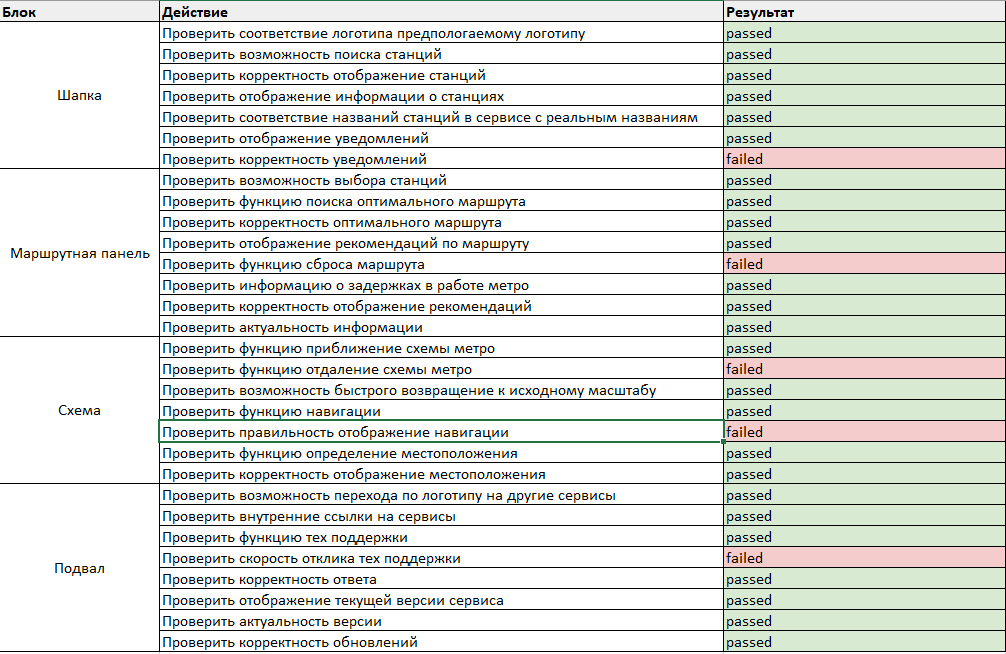
данных.

**Практическая работа номер 6.1**

Mind map



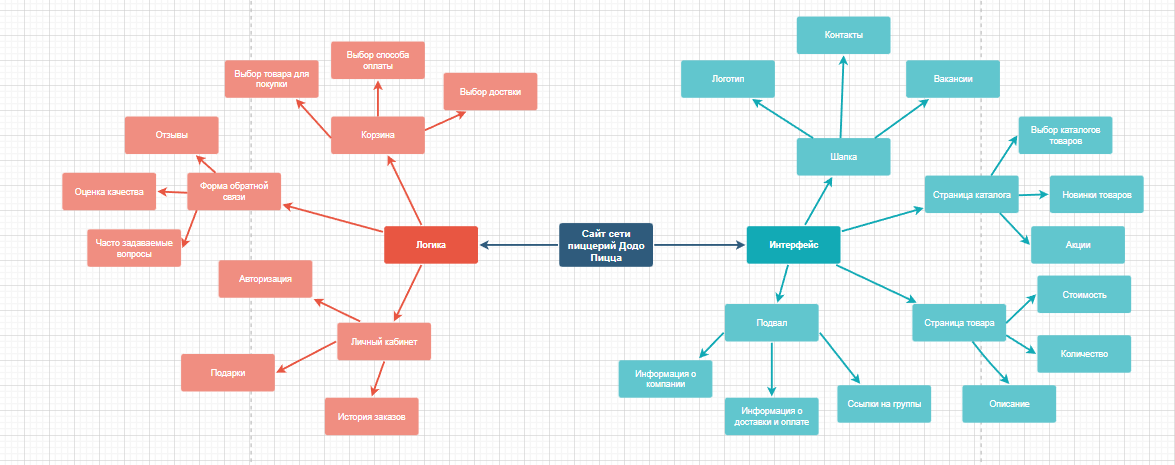
Чек лист



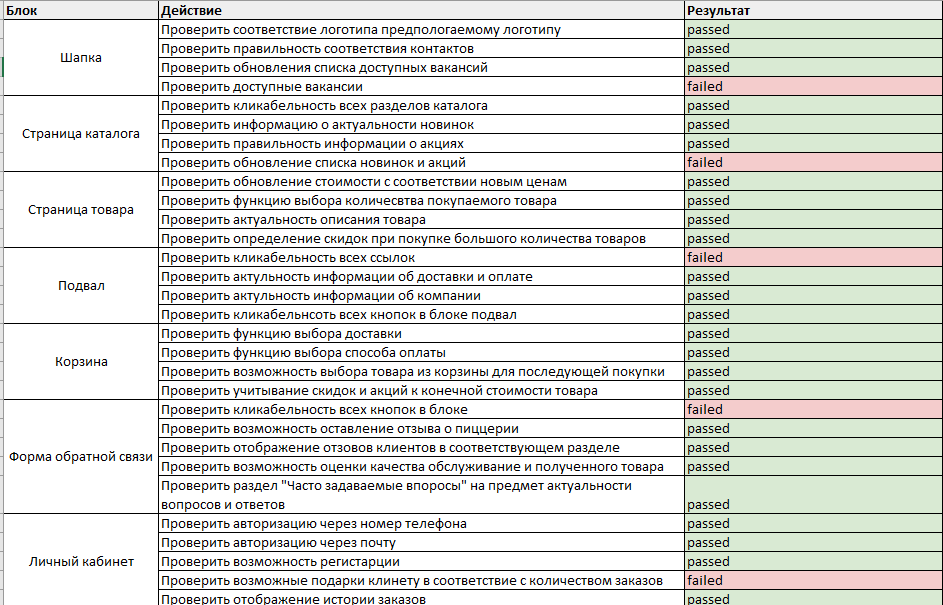
**Практическая работа номер 6.2**

Для создания mind map и чек листа был выбран сайт сети пиццерий «Додо Пицца»

Mind map

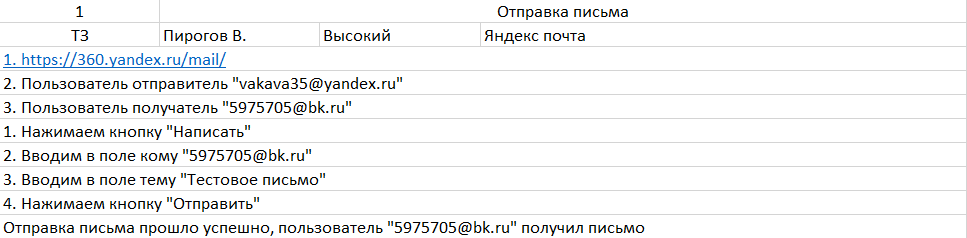


Чек лист

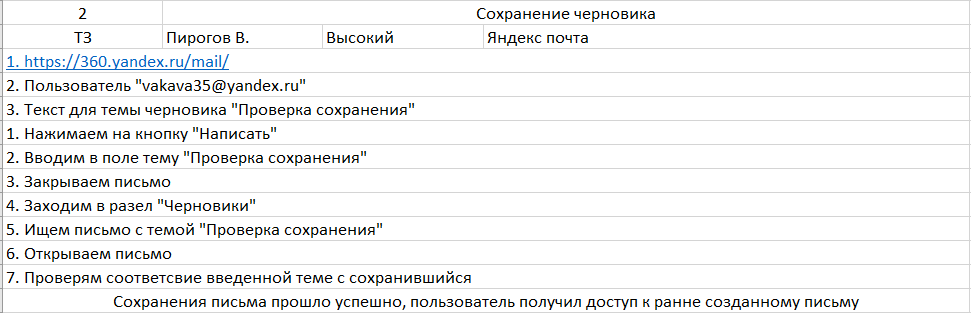


**Практическая работа номер 7**

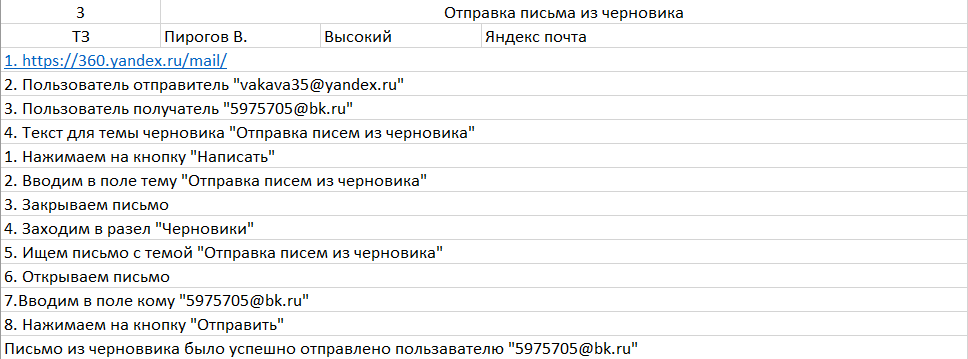
Тест-кейс «Отправка письма»



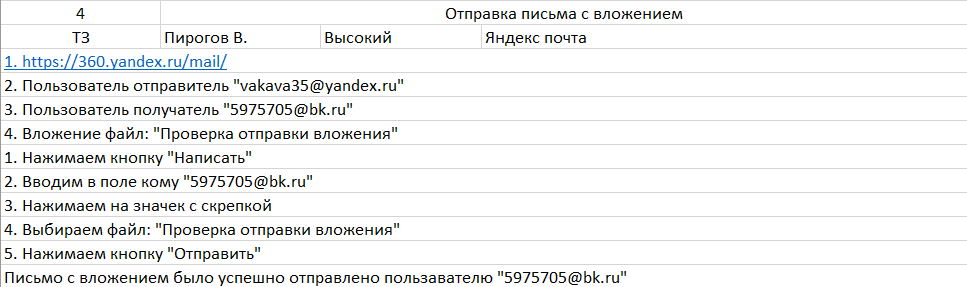
Тест-кейс «Сохранение черновика»



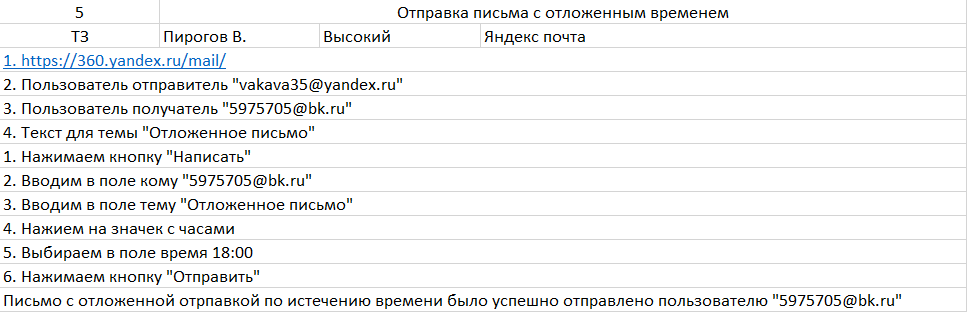
Тест-кейс «Отправка письма из черновика»



Тест-кейс «Отправка письма с вложением»



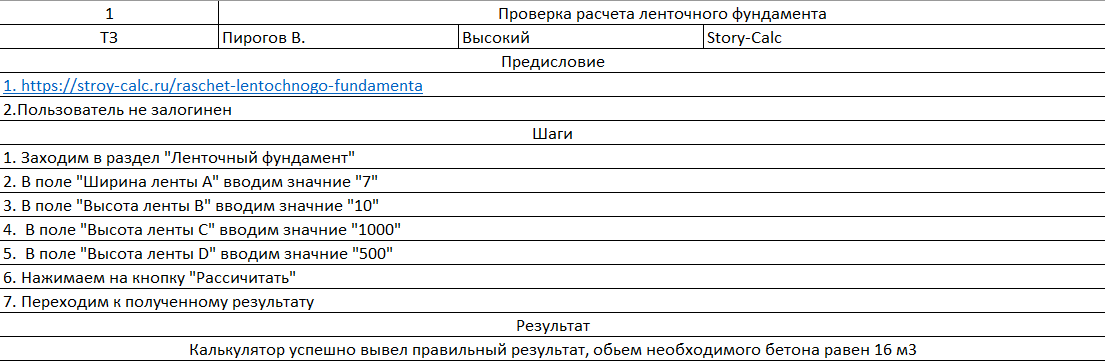
Тест-кейс «Отправка письма с отложенным временем»

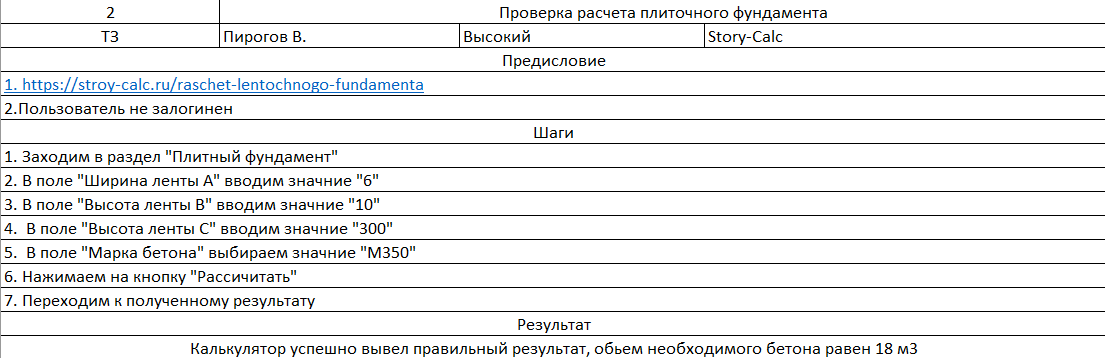


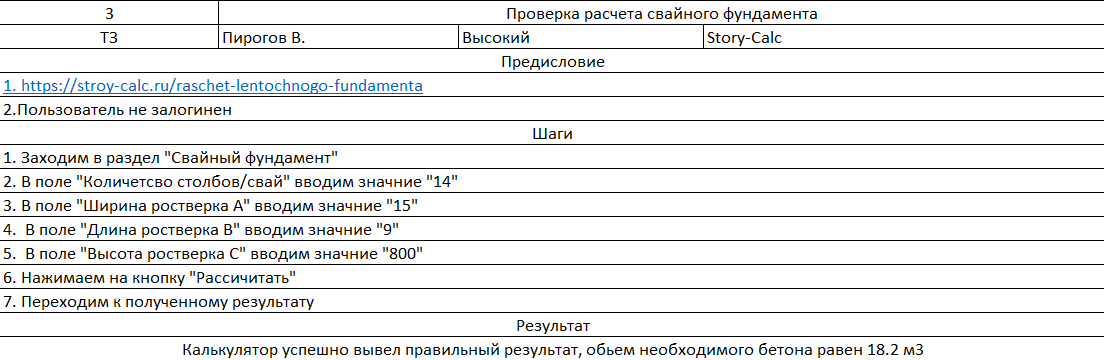
**Практическая работа номер 8**

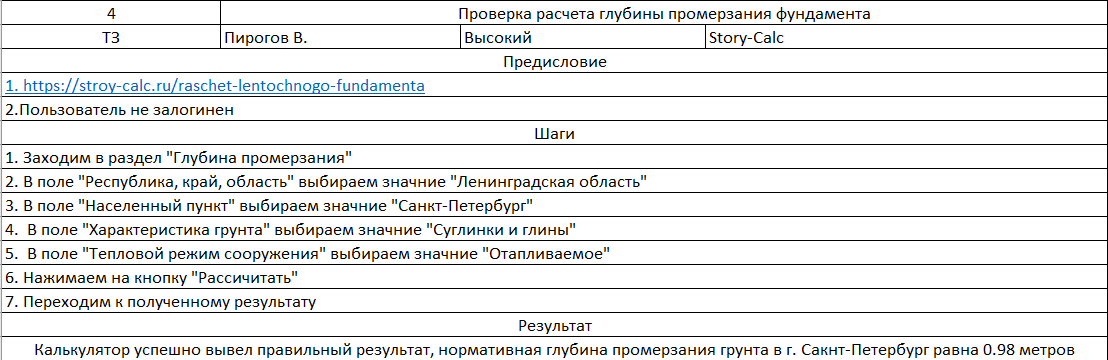
Наборы тест кейсов создавались для строительного калькулятора: [Stroy-Calc](https://stroy-calc.ru/).

Набор тест кейсов номер 1 «Проверка расчета фундамента»

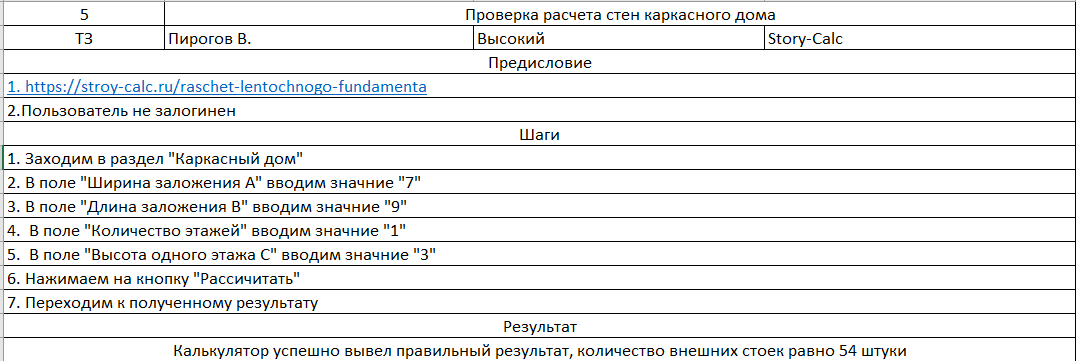


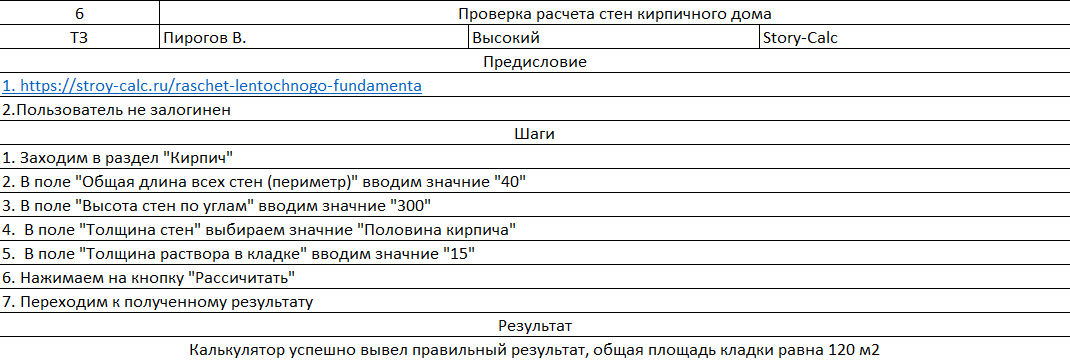


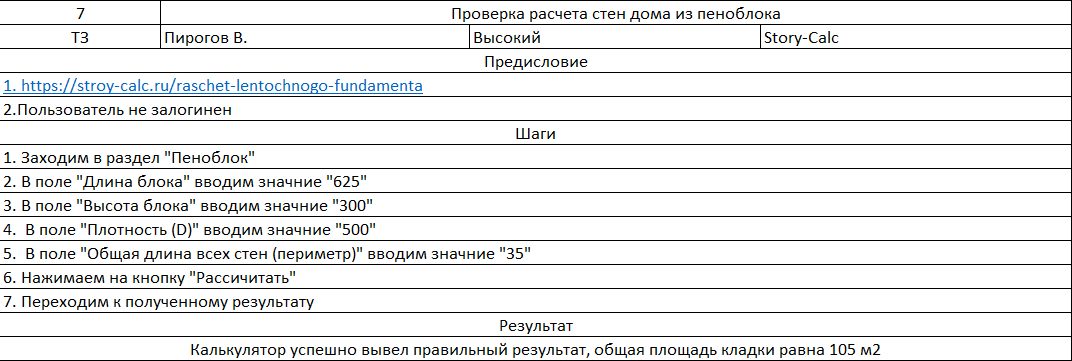


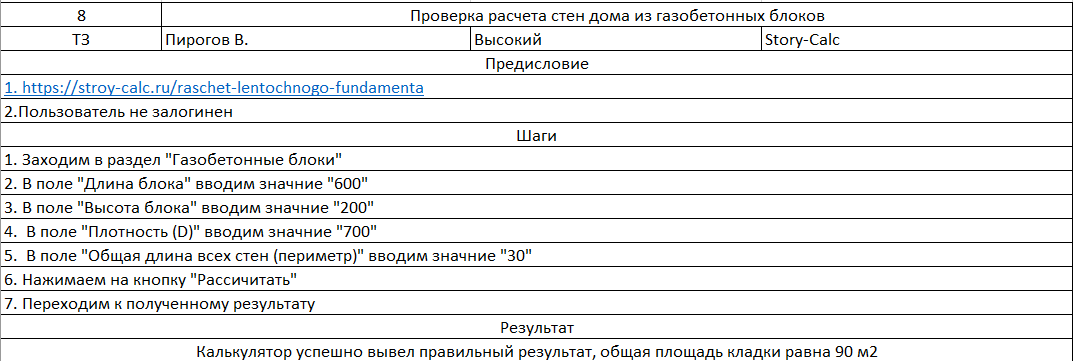


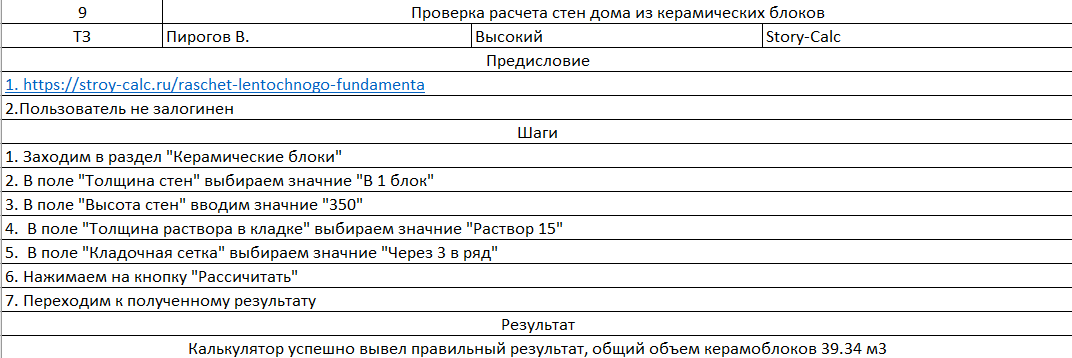
Набор тест кейсов номер 2 «Проверка расчета стен»



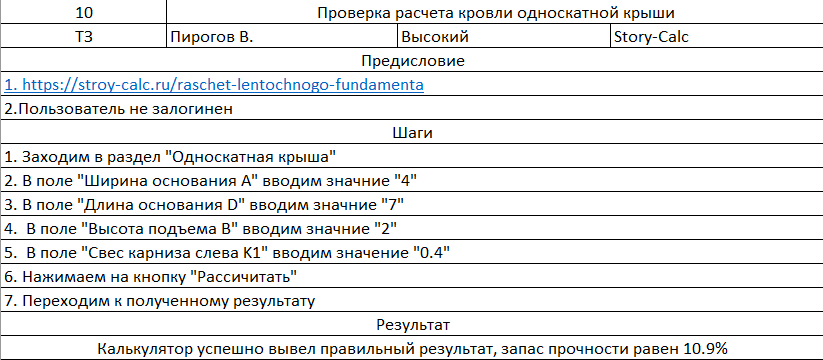


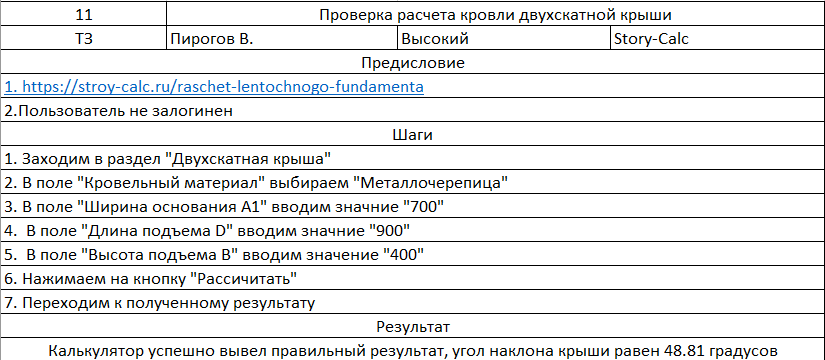




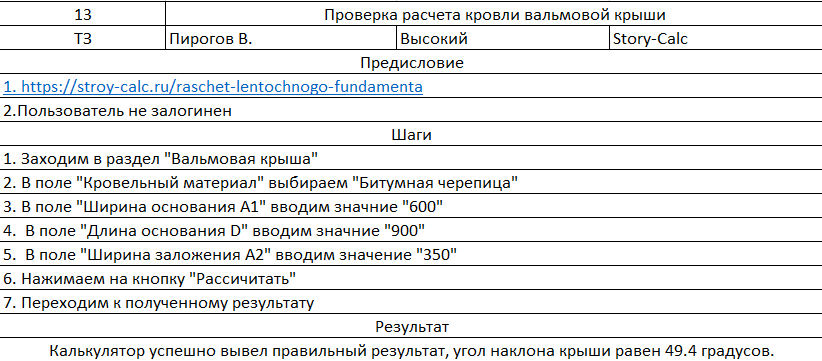


Набор тест кейсов номер 3 «Проверка расчета кровли»

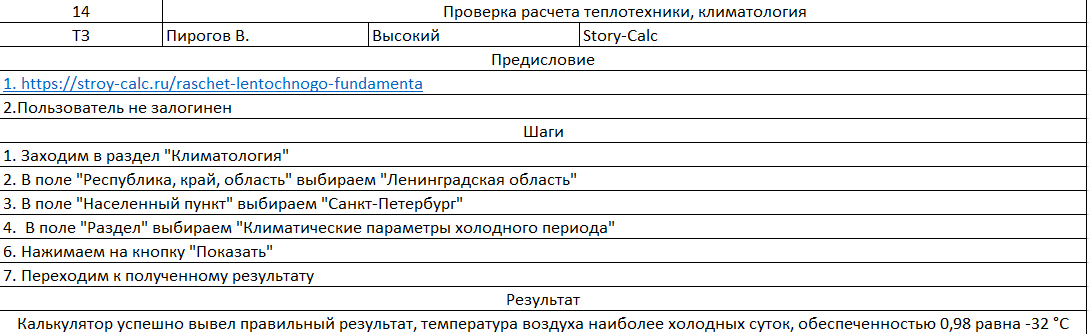


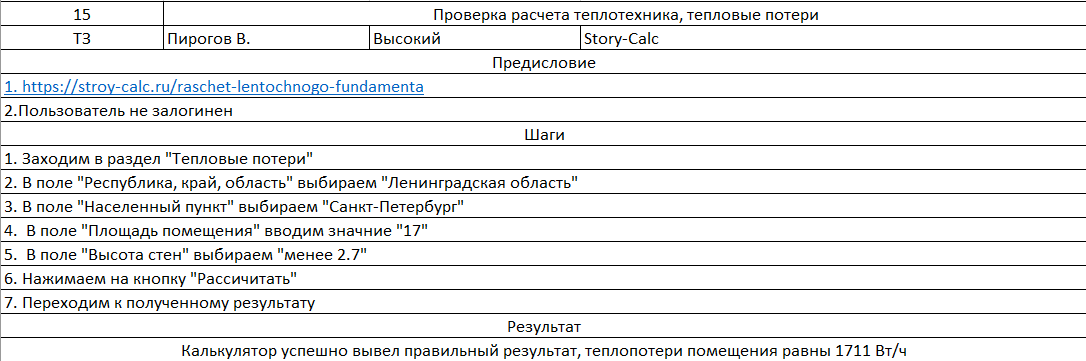


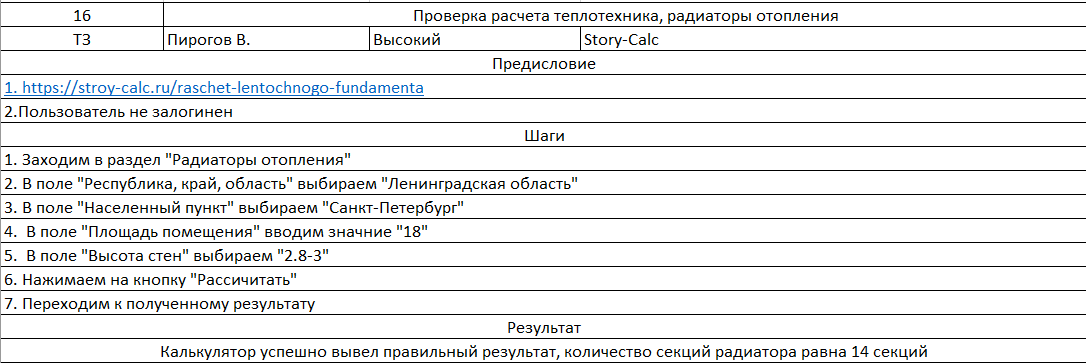


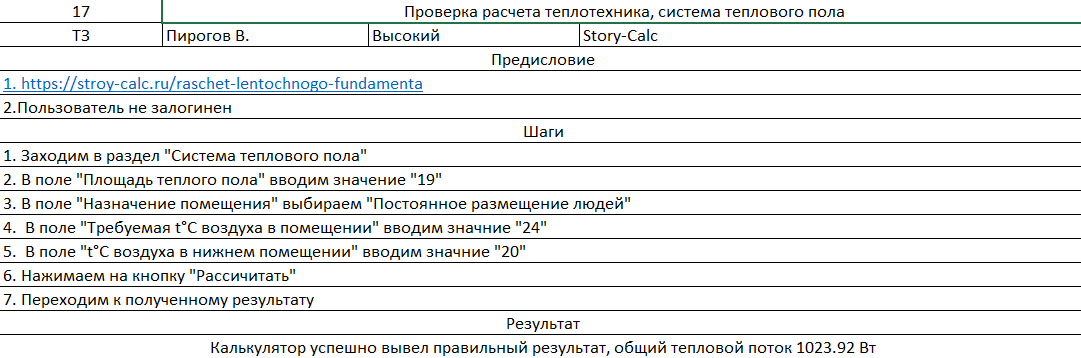


Набор тест кейсов номер 4 «Проверка расчета теплотехники»

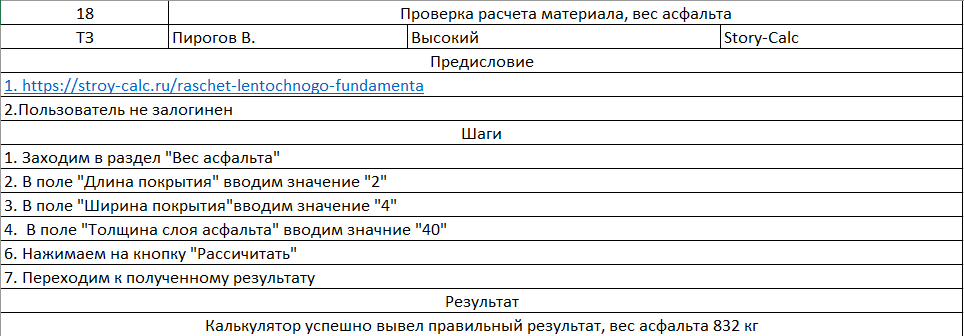


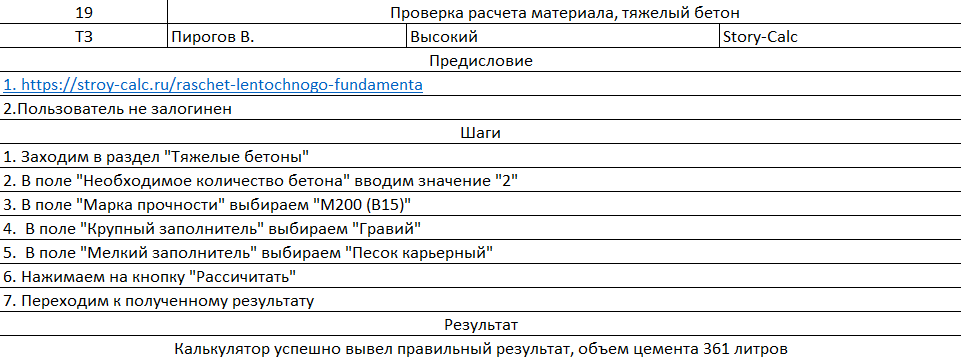


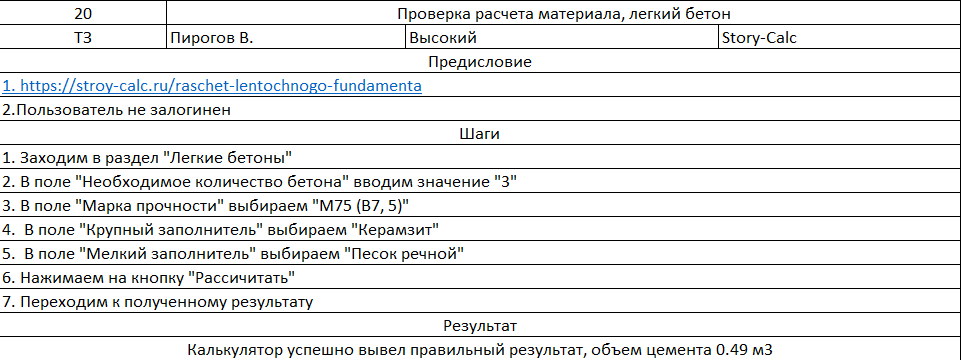


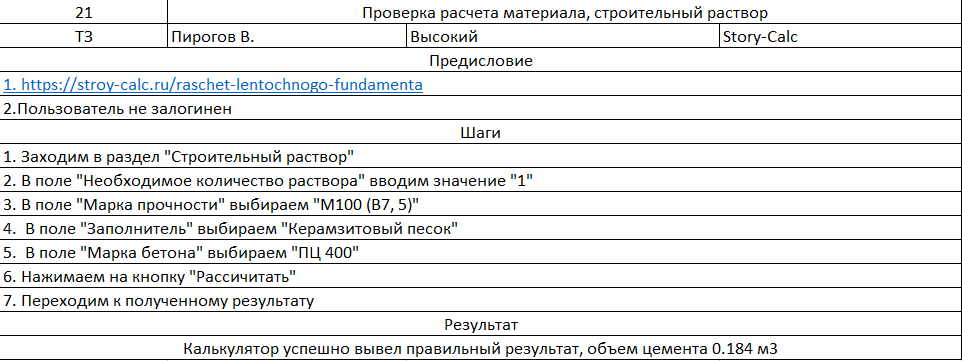


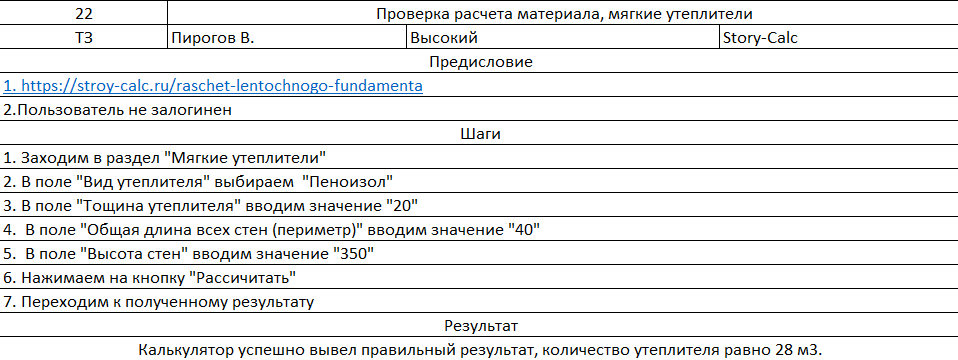
Набор тест кейсов номер 5 «Проверка расчета материалов»



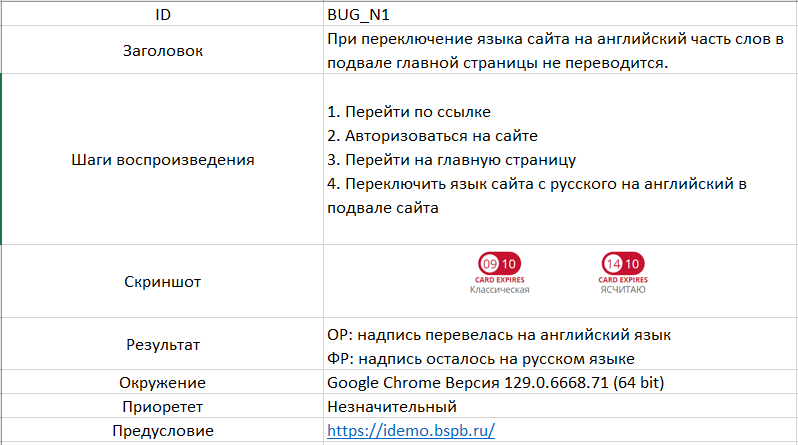


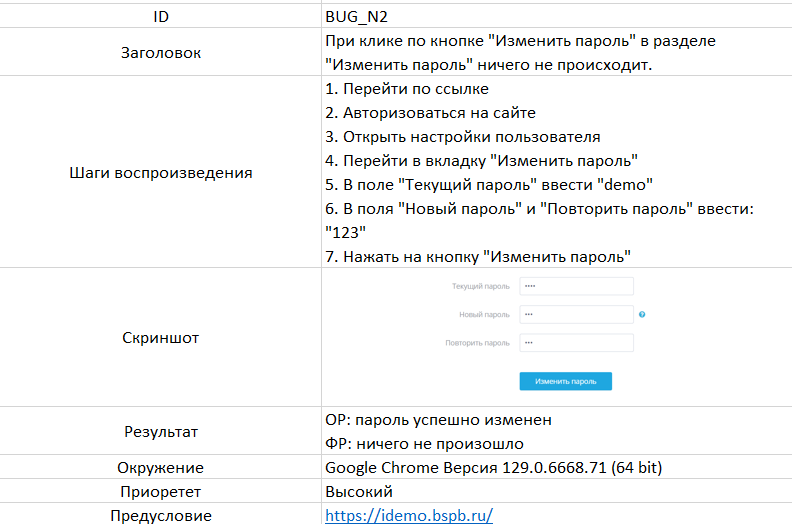


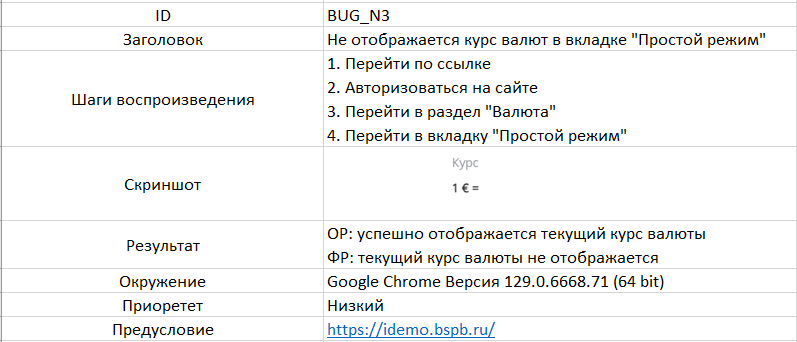


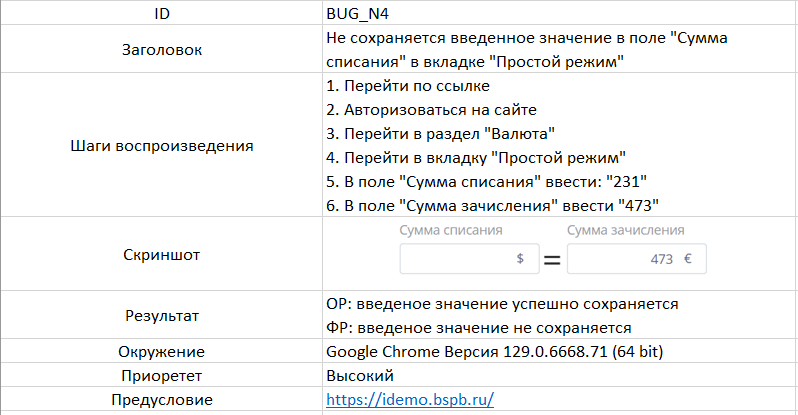


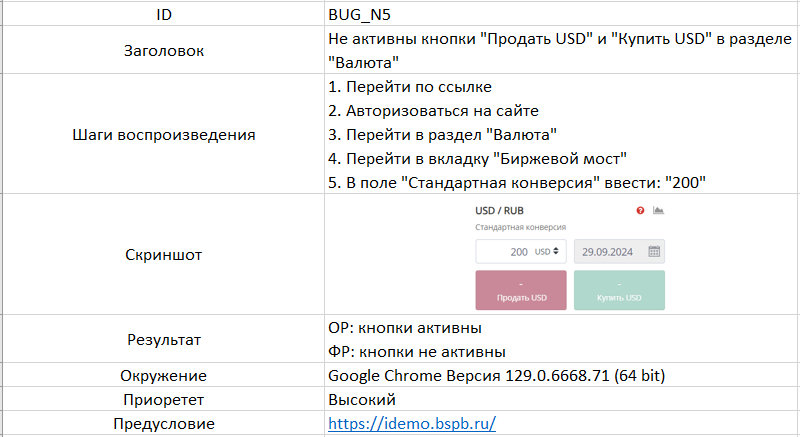
**Самостоятельная работа. Тестирование сайта и составление баг-репорта**

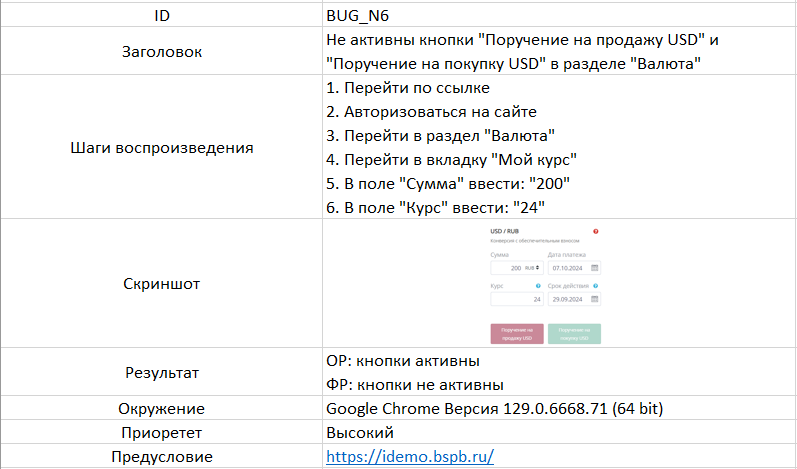


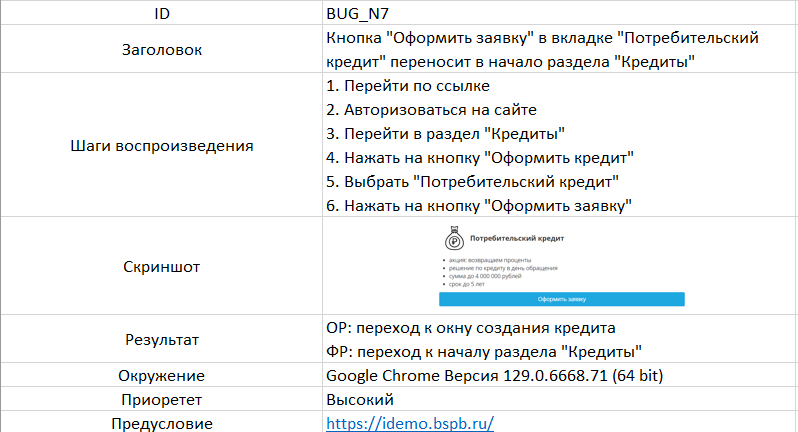


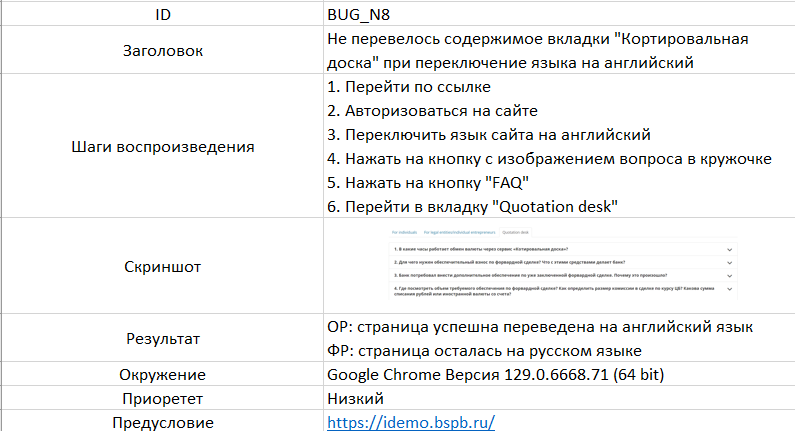


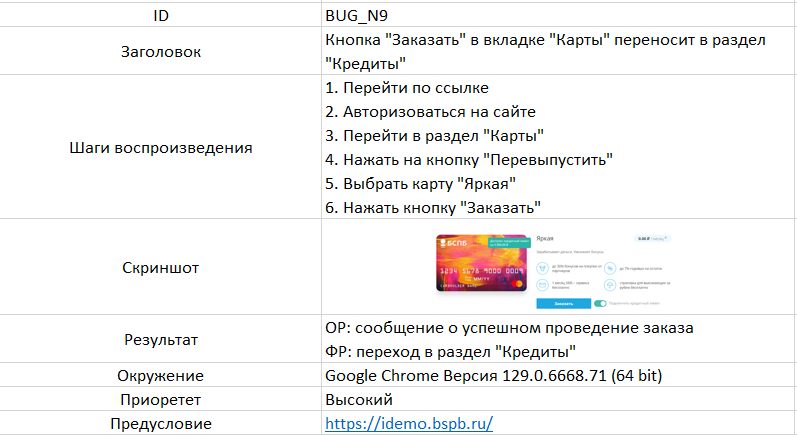


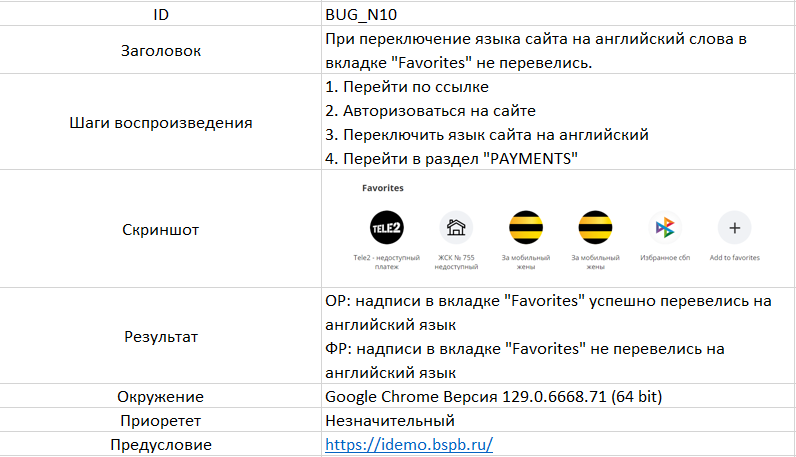


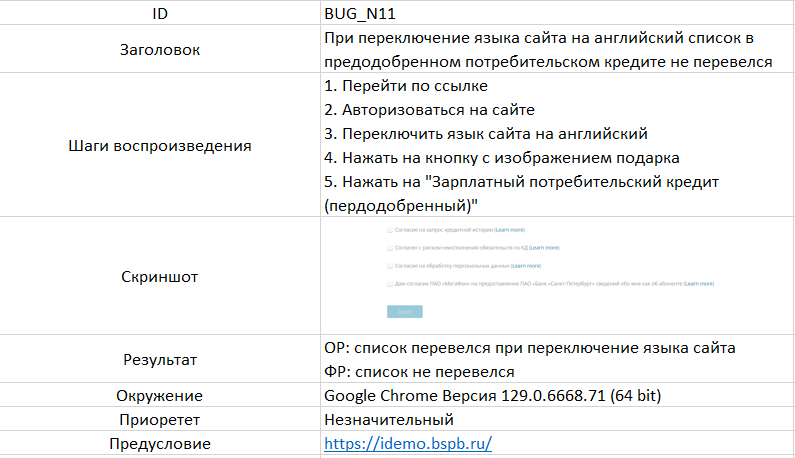


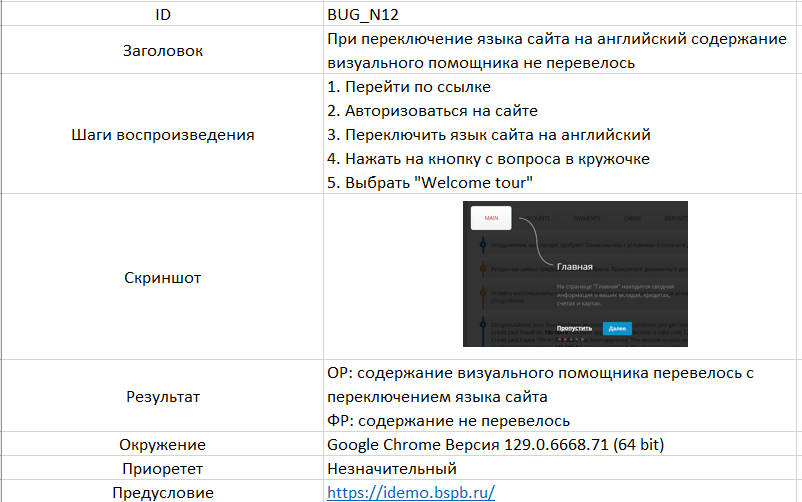


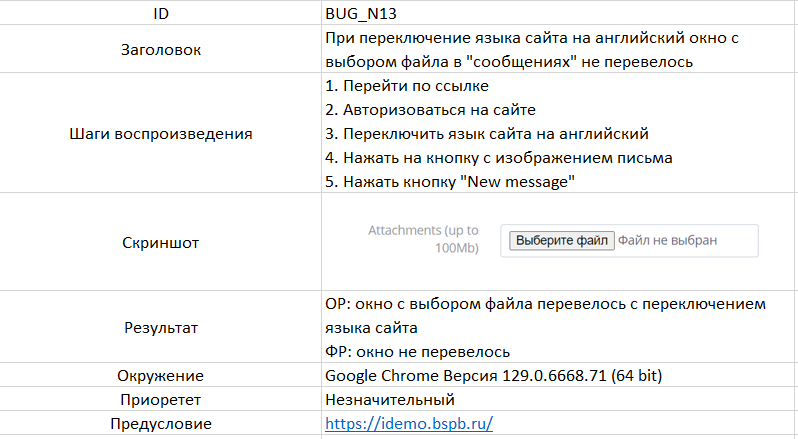


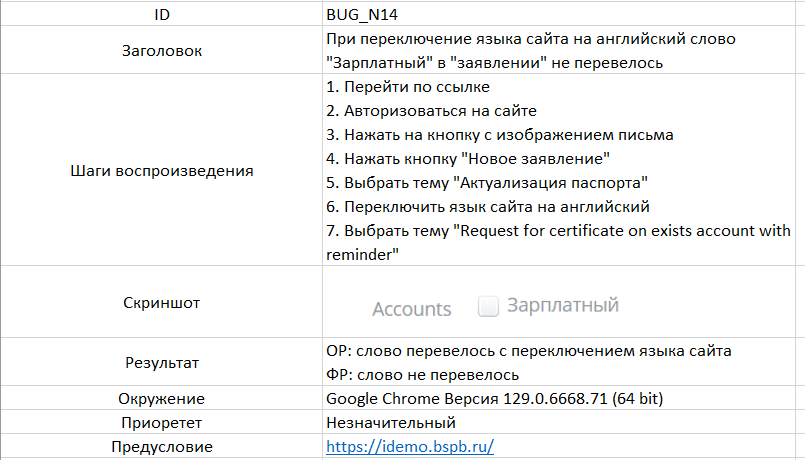


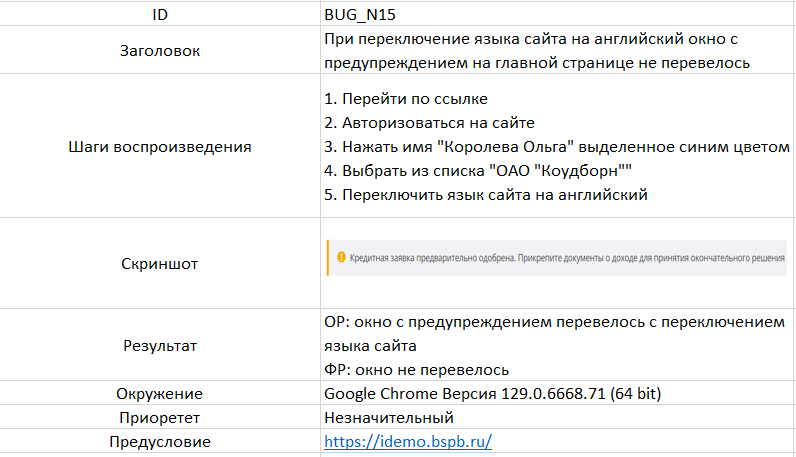




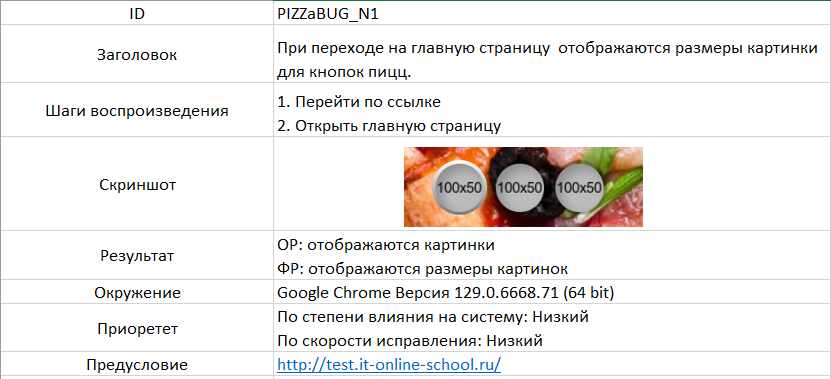


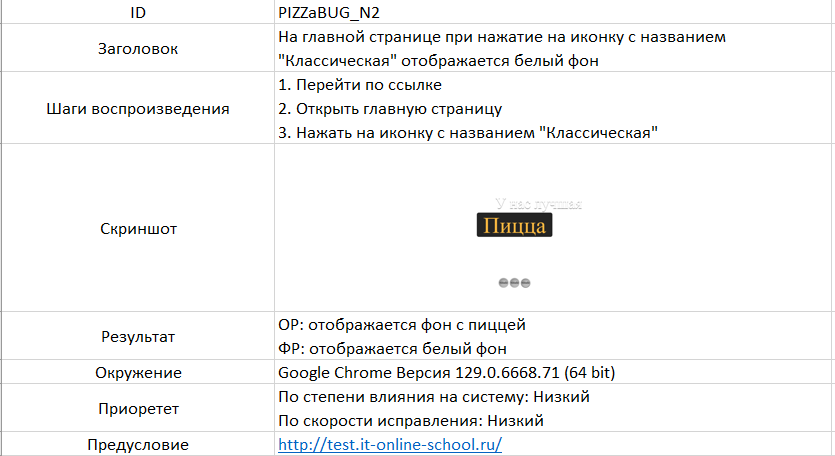


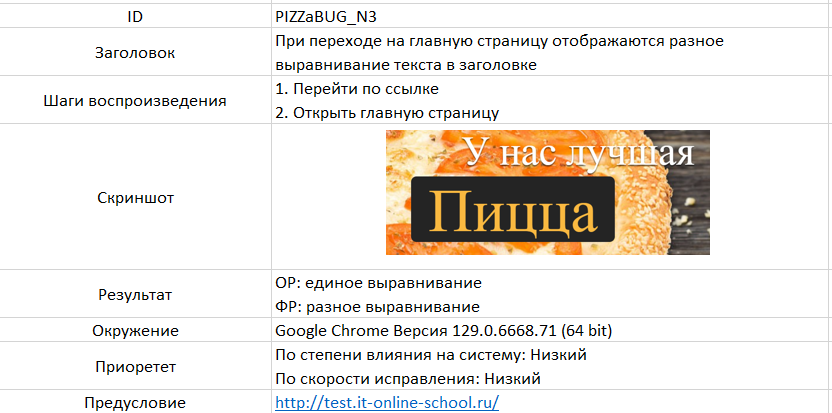


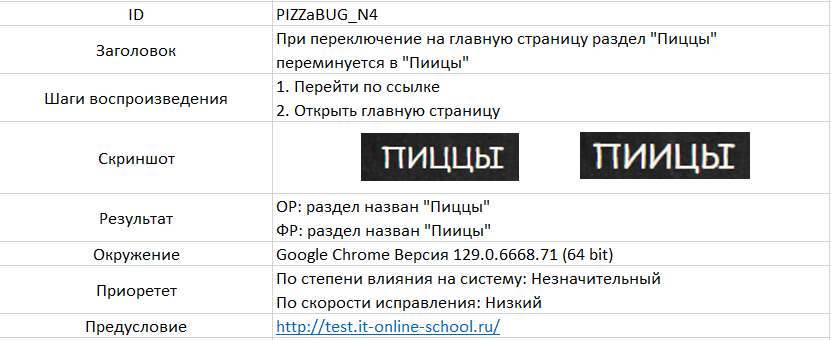


**Практическое задание 11. Баг-репортер JIRA**

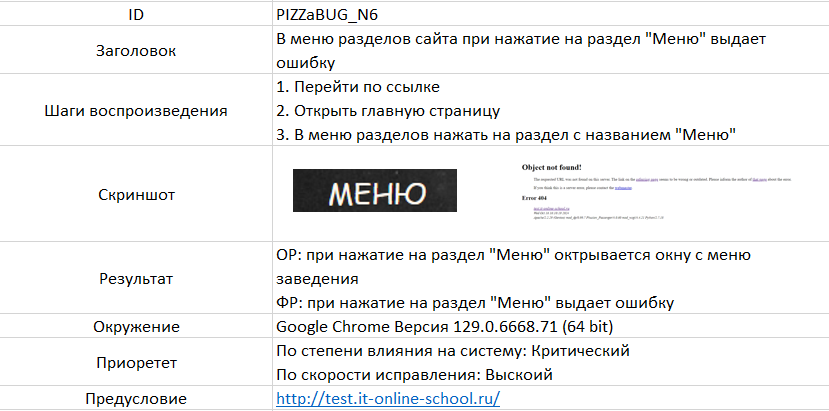


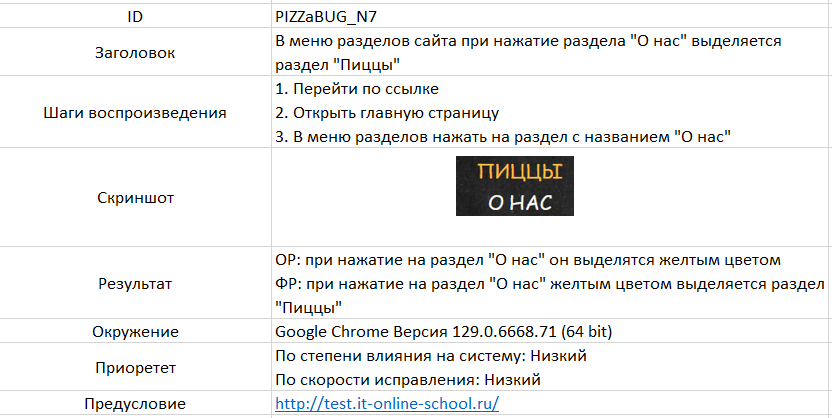


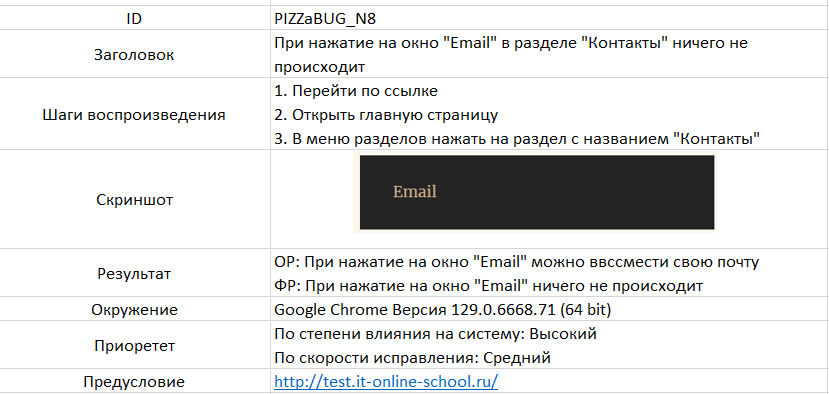


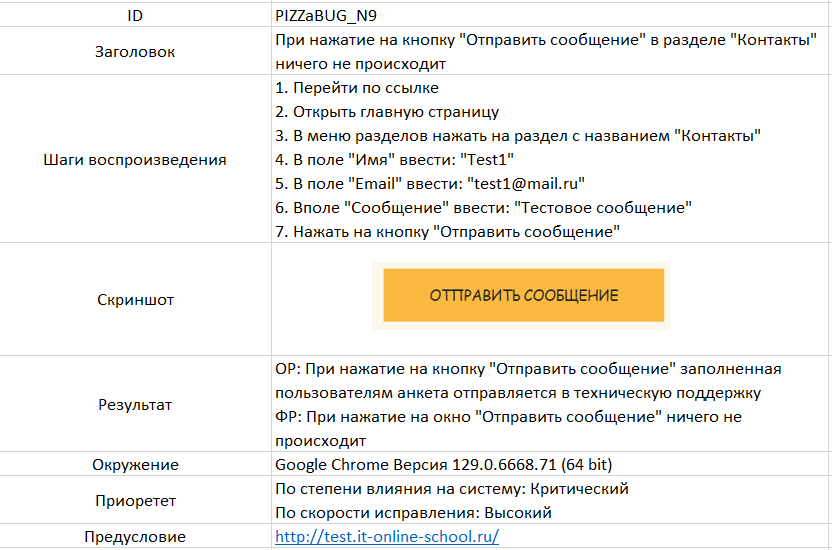


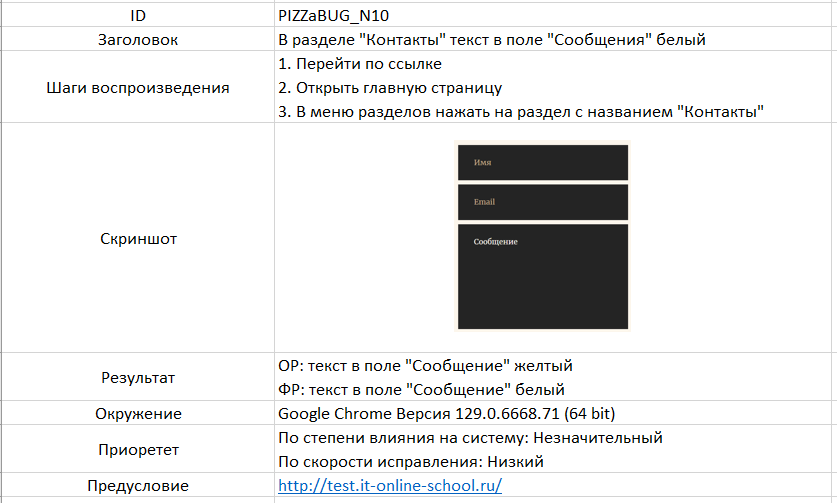


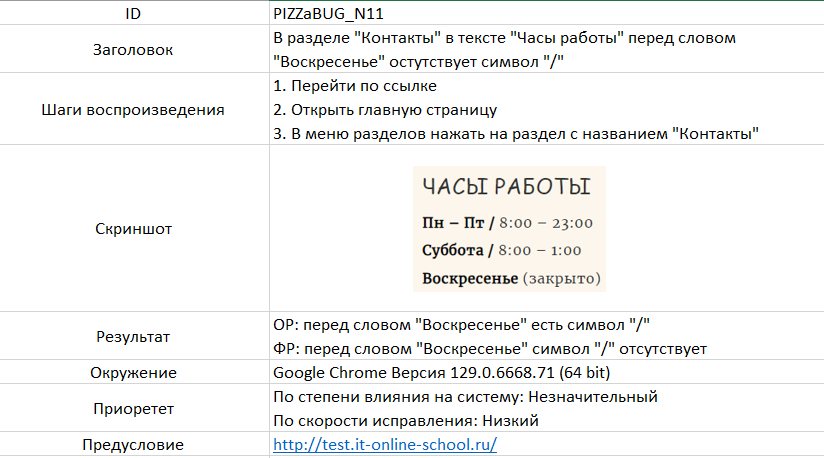


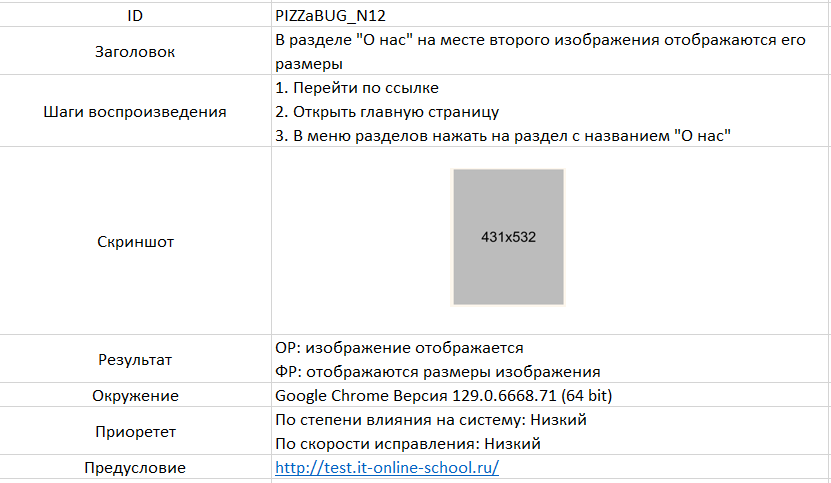


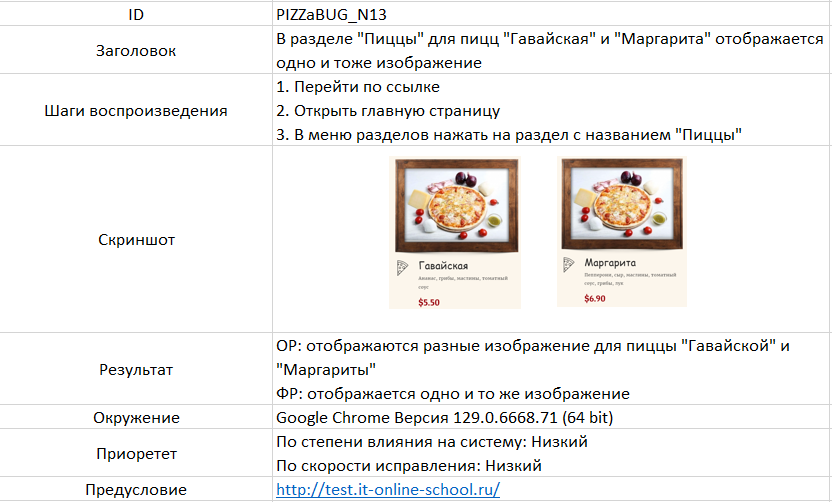


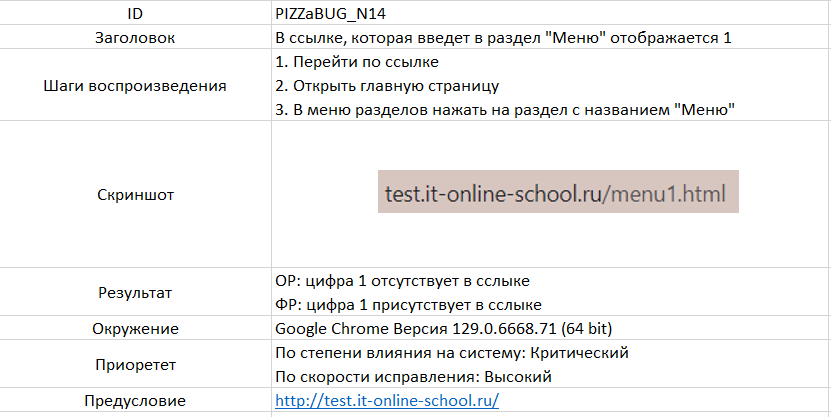


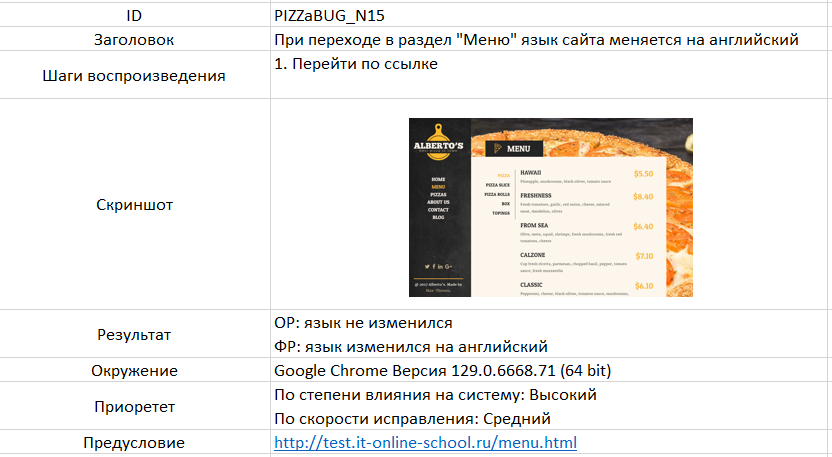


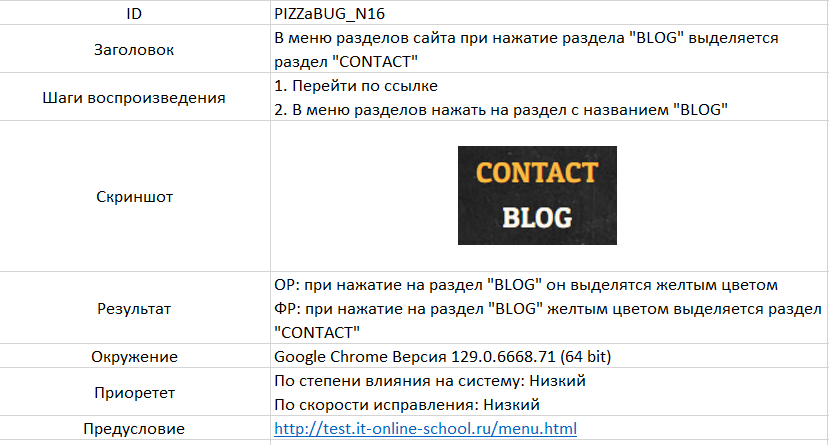


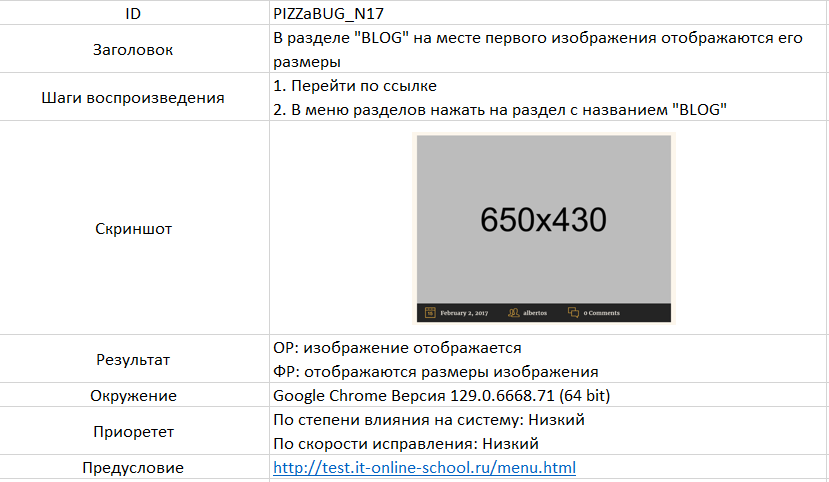


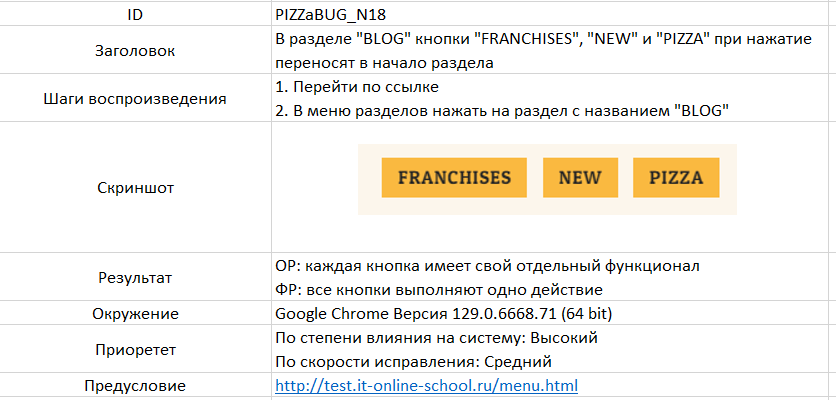


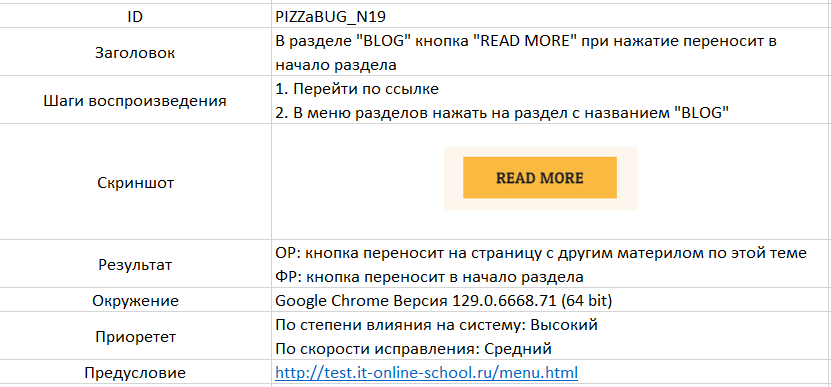


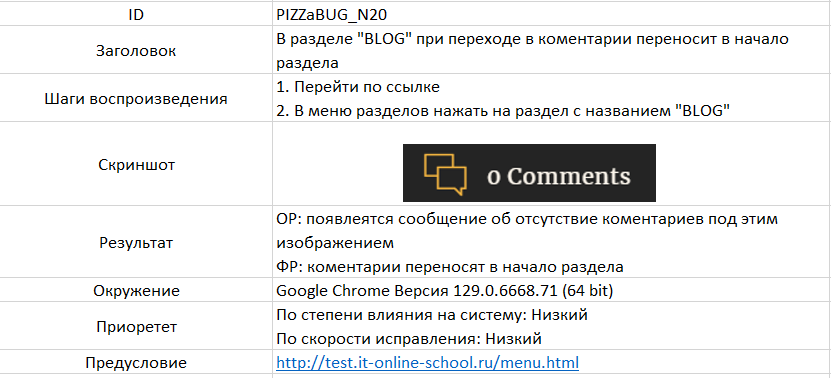




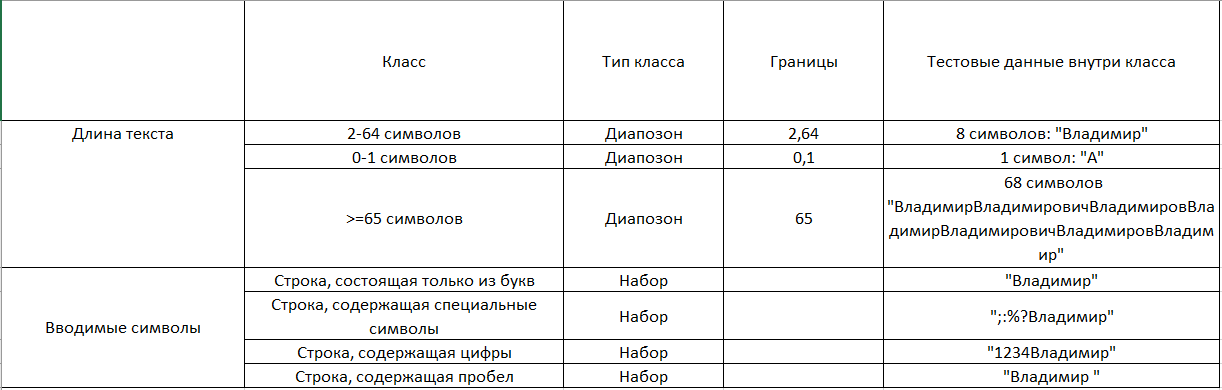




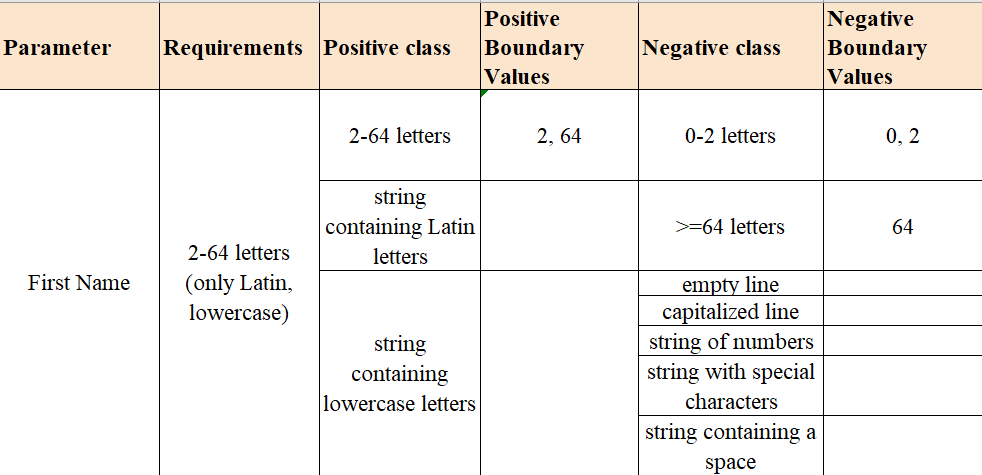


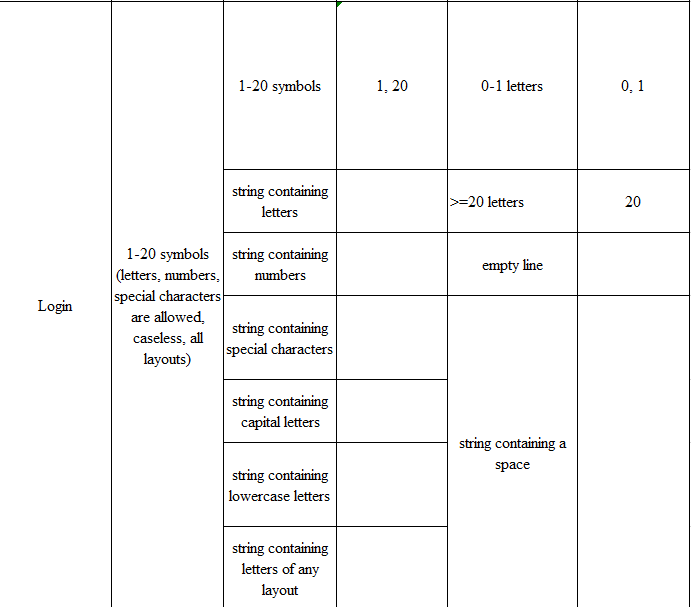


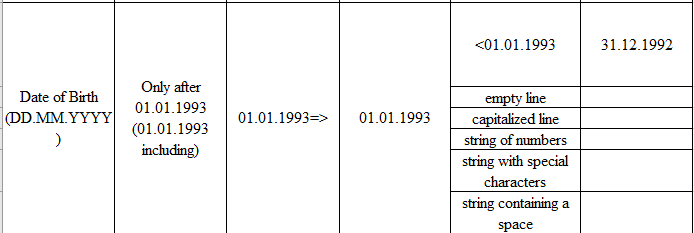
**Самостоятельная работа номер. Методики тест дизайна**.

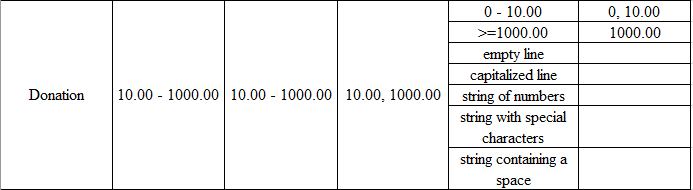


**Практическое задание 12. Тест дизайн.**

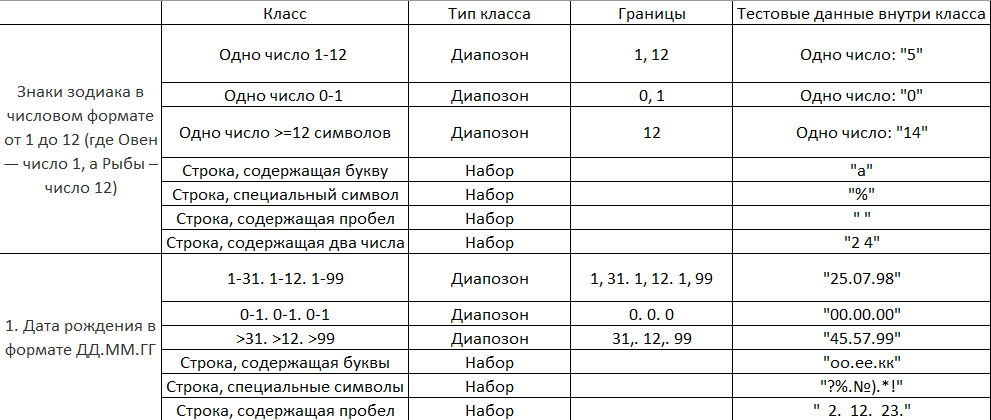




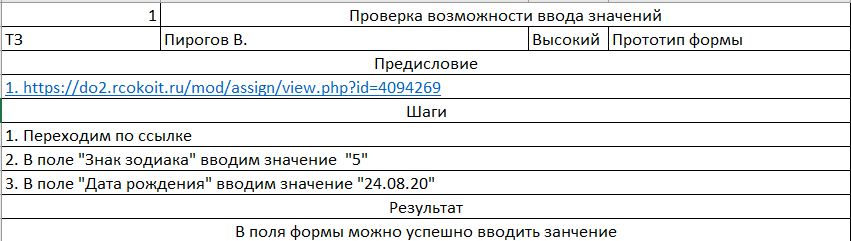


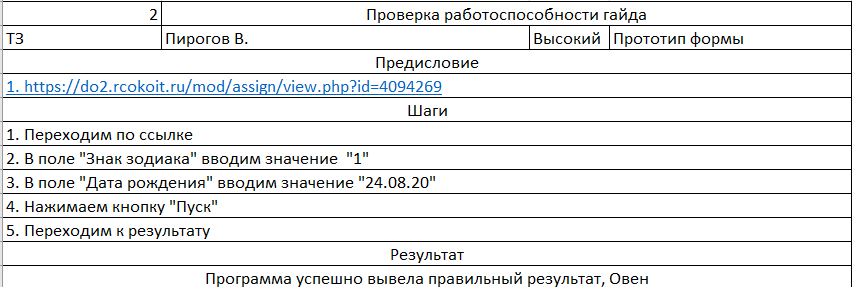


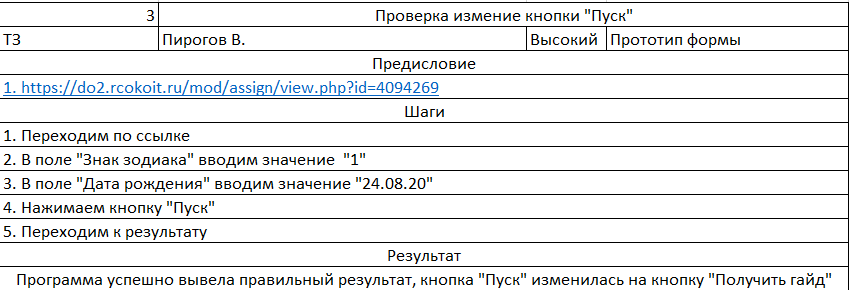
**Проверочная работа**

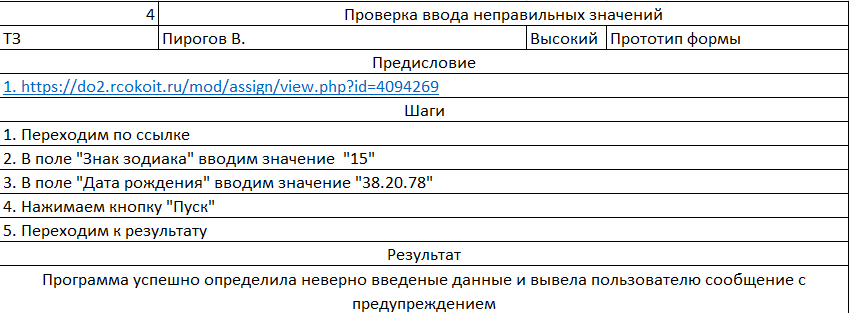


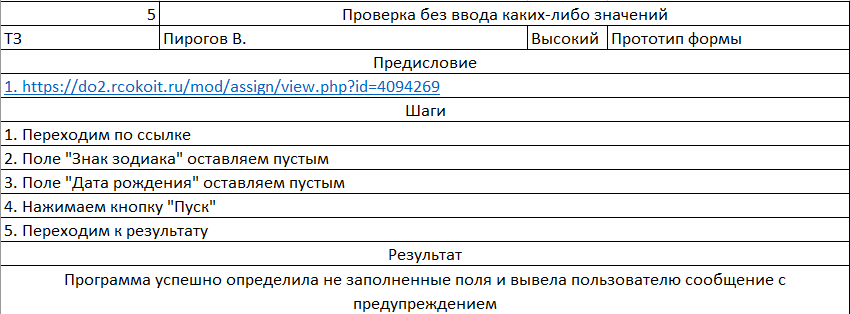
Тест кейсы







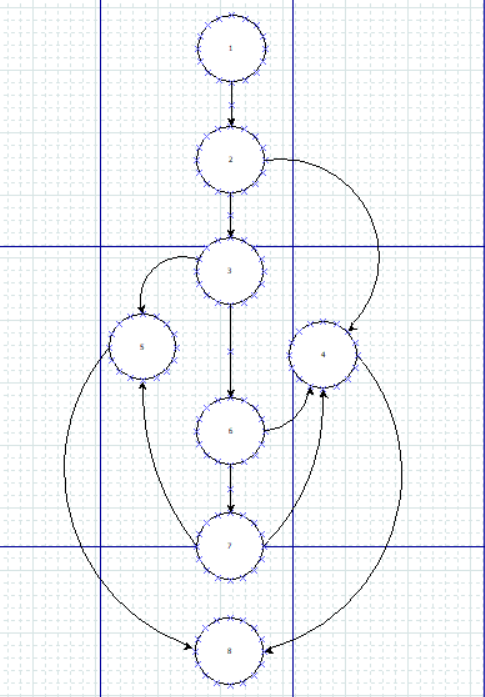




**Практическое задание 13. Тестирование базового пути**

Вариант 18: проверяет, является ли число простым

**Контрольный граф:**

****

**Этапы:**

1. int N
2. if N<2
3. if N=2
4. ENDIF
5. ENDIF
6. if (N % 2 == 0)
7. for int i = 3; i \* i <= N; i += 2
8. if Console.WriteLine(${N}является простым числом)

else Console.WriteLine(${N} не является простым числом)

**Вычисление цикломатическая сложности**

V(G) = 11 – 8 + 2 = 5

V(G) = 4 + 1 = 5

**Набор базовых путей**

Число меньше 2 (не простое).

Путь 1: (1) → (2) → (4) → (8)

Число равно 2 (простое).

Путь 2: (1) → (2) → (3) → (5) → (8)

Число четное и больше 2 (не простое).

Путь 3: (1) → (2) → (3) → (6) → (4) → (8)

Число нечетное и делится хотя бы на одно нечетное число (не простое).

Путь 4: (1) → (2) → (3) → (6) → (7) → (4) → (8)

Число нечетное и не делится ни на одно нечетное число до √N (простое).

Путь 5: (1) → (2) → (3) → (6) → (7) → (5) → (8)

**Генерация тестовых случаев для каждого пути**

Путь 1. Пример: 1

Путь 2. Пример: 2

Путь 3. Пример: 8

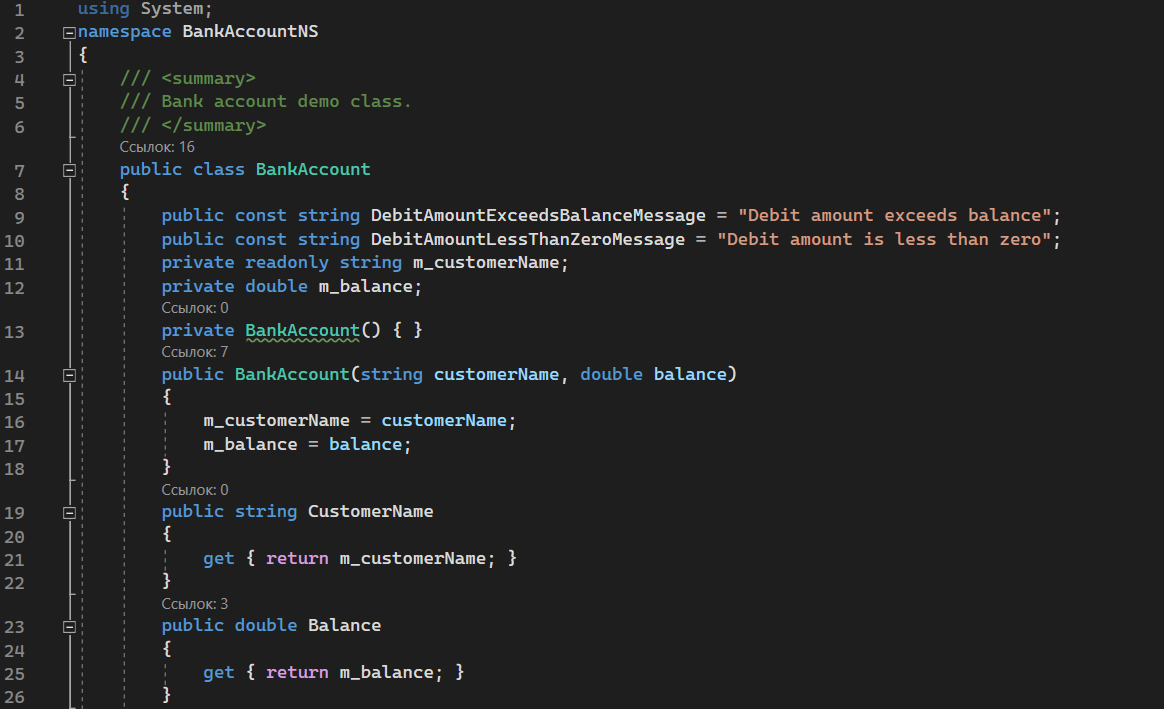
Путь 4. Пример: 8

Путь 5. Пример: **√**7

**Практическое задание 16. Unit Test**

Задание: выполнить задание по образцу и добавить 3 своих теста

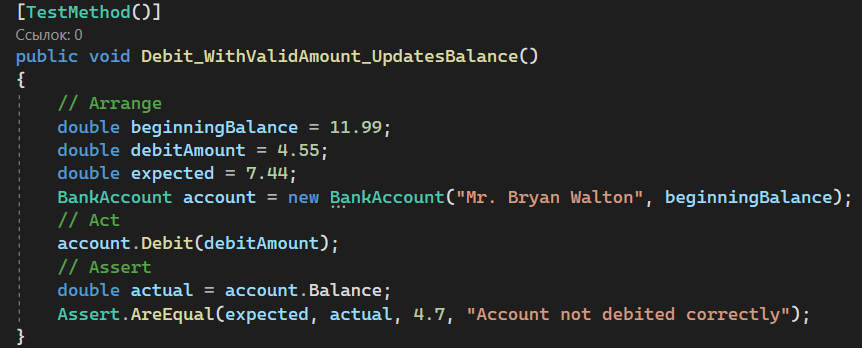
**Основной код (BankAccount):**



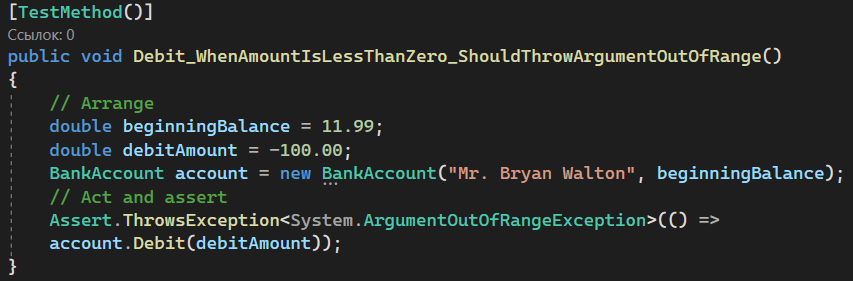


**Код тестов, сделанных по образцу:**

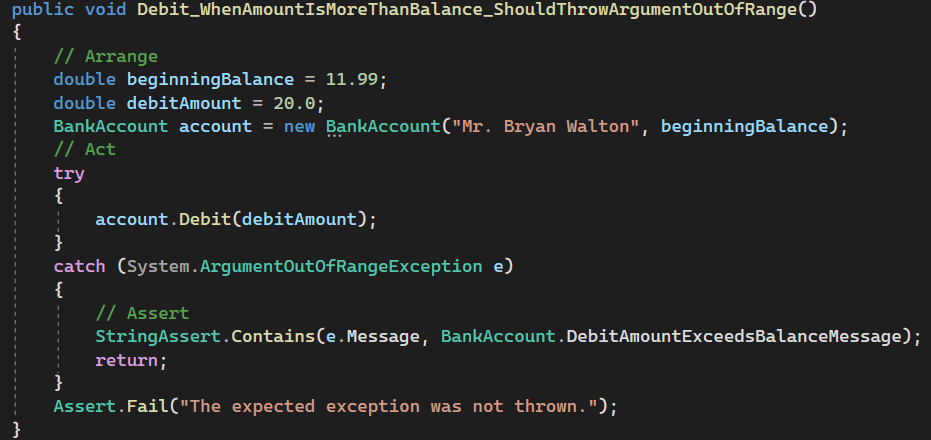
Тест 1. Проверка, снимается ли со счета нужная сумма при допустимом размере кредита (со значением меньшим, чем баланс счета, и большим, чем ноль).



Тест 2. Проверки правильного поведения системы в случае, когда сумма по дебету меньше нуля.

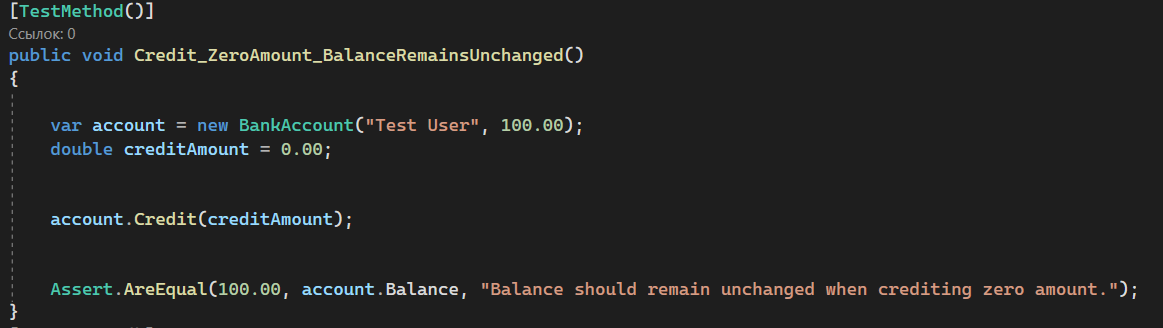


Тест 3. Проверки правильного поведения системы в случае, когда сумма по дебету больше нуля.

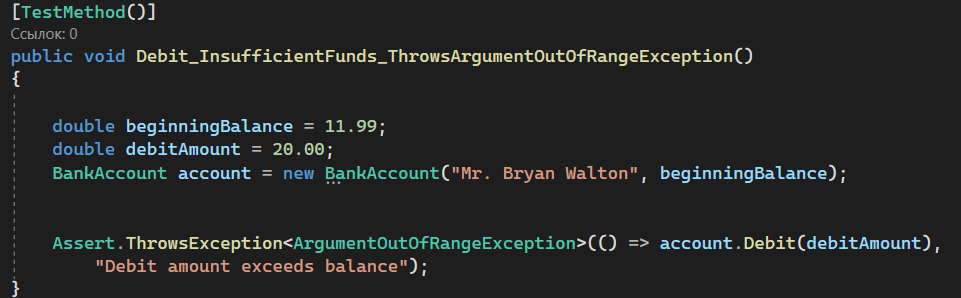


**Свои тесты:**

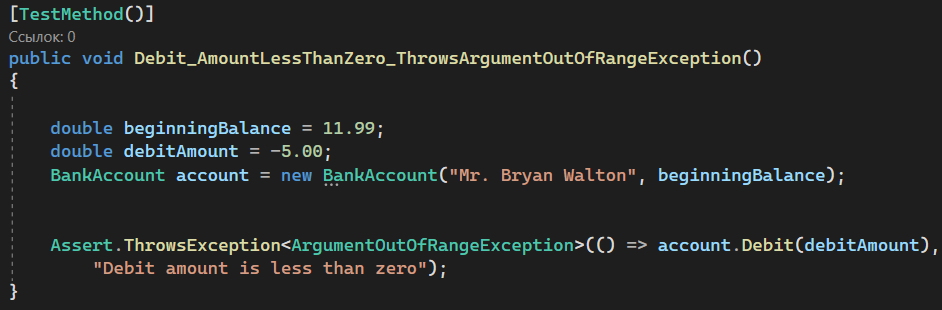
Тест 4. Проверка на зачисление нулевой суммы



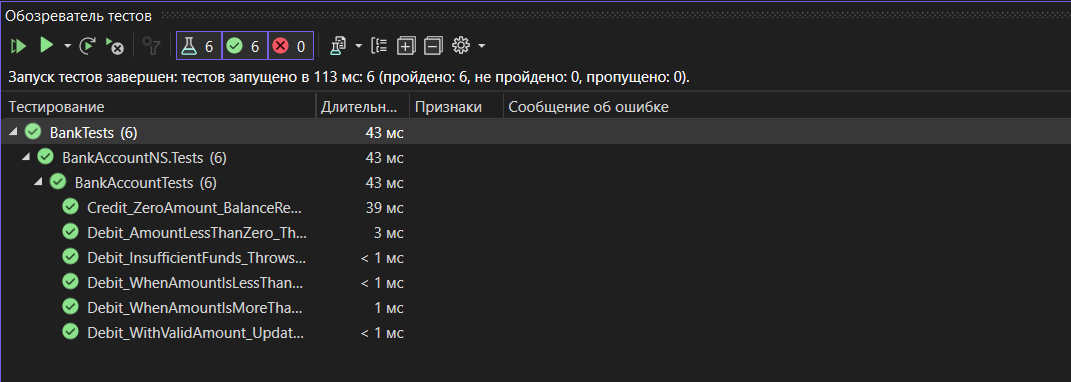
Тест 5. Проверка, что при недостаточном остатке на балансе метод выводит исключение ArgumentOutOfRangeException с сообщением.



Тест 6. Проверка, что при отрицательном значении суммы списания метод выводит исключение ArgumentOutOfRangeException с сообщением.



**Проверка выполнение всех 6 тестов**



**Практическое задание 17. Unit Test**

Задание:

– загрузить проект на свою машину и изучить его самостоятельно;

– определить и описать назначение данного проекта;

– произвести анализ приведенных классов, содержащих модульные тесты (FileTest, FileStorageTest), а также тестовых классов (File, FileStorage);

– исправить модульные тесты, имеющиеся ошибки;

– дополнить имеющиеся тесты своими в количестве 3 штук.

**Назначение данного проекта**

Данный проект представляет собой имитацию файлового хранилища в памяти. Он позволяет сохранять, удалять, проверять наличие и получать файлы. Хранилище имеет ограниченный размер и проверяет уникальность имен файлов. При добавлении файла проверяется, есть ли место и не существует ли файл с таким же именем. Вся работа с файлами ведется в памяти, без обращения к файловой системе.

**Анализ приведенных классов**

Класс «FileStorage»

Это класс, который моделирует файловое хранилище в памяти. Он предоставляет функционал для записи файлов, проверки их существования, удаления и получения. У хранилища есть ограниченный размер, и при добавлении файлов проверяется наличие свободного места и уникальность имени файла.

Класс «FileStorageTest»

1. Тест «WriteTest(File file)»

Описание: тестирует успешную запись файла в хранилище.

1. Тест «WriteExceptionTest(File file)»

Описание: тестирует ситуацию, когда добавляется дублирующийся файл, проверяя что вызывается исключение.

1. Тест «IsExistsTest(File file)»

Описание: тестирует метод IsExists, проверяя, что он корректно определяет наличие файла в хранилище.

1. Тест «DeleteTest(File file, String fileName**)**»

Описание: тестирует метод Delete для удаления файла.

1. Тест «GetFilesTest()»

Описание: тестирует метод GetFiles, проверяя, что он возвращает список, в котором каждый элемент не равен null.

1. Тест «GetFileTest(File expectedFile)»

Описание: тестирует метод GetFile для получения файла по имени.

Класс «File»

Данный класс представляет собой простой объект для хранения информации о файлах, предназначенный для использования в различных контекстах.

Класс «FileTest»

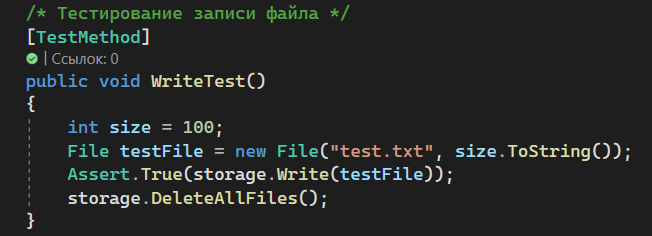
Тест «GetSizeTest(File newFile, String name, String content)»

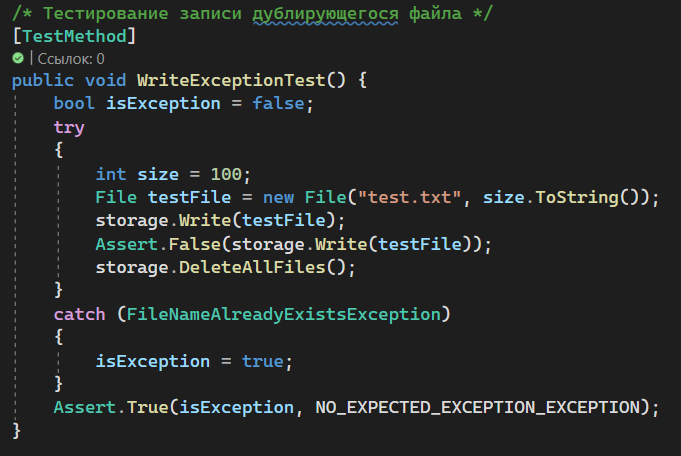
Описание: тестирует метод GetSize класса File, проверяя корректность вычисления размера файла.

Тест «GetFilenameTest(File newFile, String name, String content)»

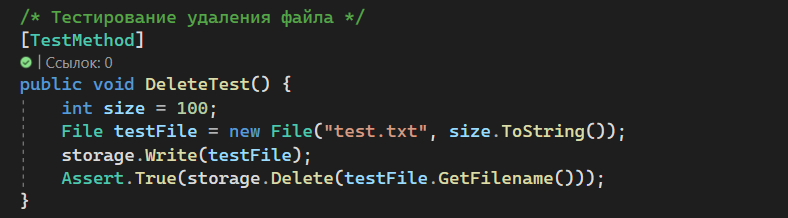
Описание: тестирует метод GetFilename класса File, проверяя корректность получения имени файла.

**Исправленные тесты**

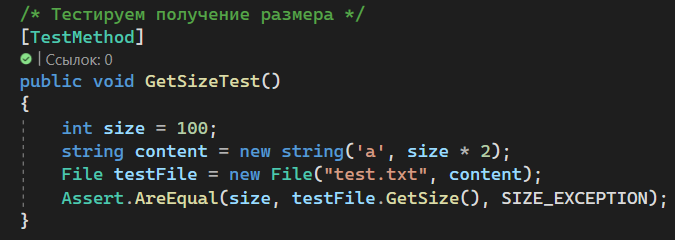
****

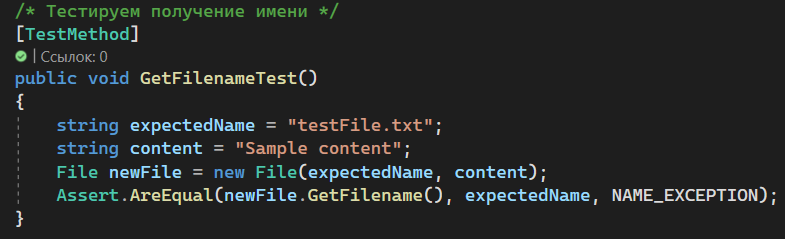
****

****

****

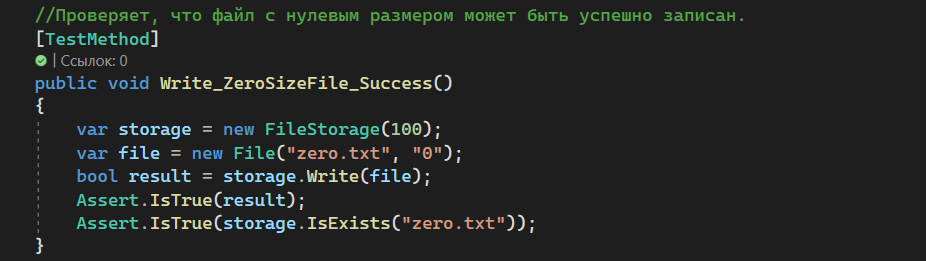
****

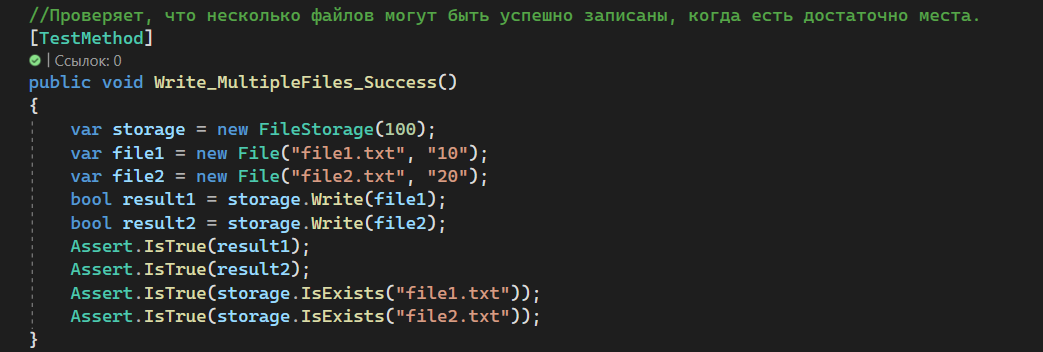
****

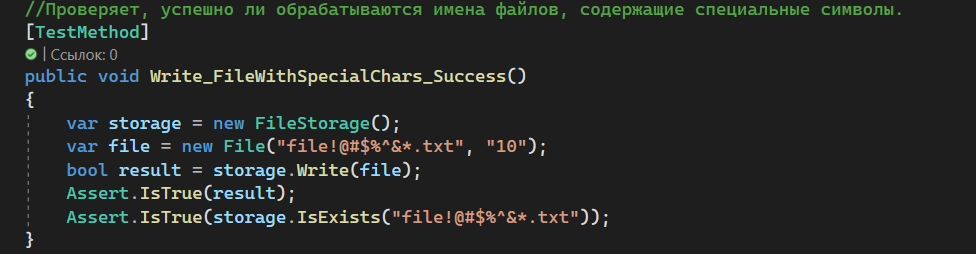
****

Из 8 тестов было исправлено 7.

**Свои тесты**

****

****

****

**Все тесты работают**

