МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт Информационных технологий (ИТ)

Кафедра Математического обеспечения и стандартизации информационных технологий (МОСИТ)

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 4

по дисциплине

«Тестирование и верификация программного обеспечения»

Выполнил:
студент группы ИКБО-36-22 Шумахер М.Е.

Проверил:
Доцент Чернов Е.А.

Москва 2024

СОДЕРЖАНИЕ

1 ЗАДАНИЕ 3
2 РАЗРАБОТКА ПЛАНА ТЕСТИРОВАНИЯ 4
2.1 Идентификатор тестового плана 4
2.2 Ссылки на используемые документы 4
2.3 Введение 4
2.4 Тестируемые элементы 4
2.5 Проблемы риска тестирования ПП 4
2.6 Особенности или свойства, подлежащие тестированию 4
2.7 Особенности (свойства), не подлежащие тестированию 4
2.8 Подход 4
2.9 Критерии смоук-тестирования 5
2.10 Критерии прохождения тестов 5
2.11 Критерии приостановки и возобновления работ 5
2.12 Тестовая документация 5
2.13 Основные задачи тестирования 5
2.14 Необходимый персонал и обучение 5
2.15 Требования среды 5
2.16 Распределение ответственности 6
2.17 График работ (календарный план) 6
2.18 Риски и непредвиденные обстоятельства 6
2.19 Утверждение плана тестирования 6
2.20 Глоссарий 6
3 АВТОМАТИЗАЦИЯ ТЕСТИРОВАНИЯ WEB-Приложения 7
3.1 Тест 1 7
3.2 Тест 2 7
3.3 Тест 3 8
4 ВЫВОД 10

1 ЗАДАНИЕ

Необходимо разработать план тестирования для приложения, созданного в практической работе №3, а также провести автоматизированное тестирование Web-приложения. В отчете должны быть описаны этапы разработки плана тестирования, а также подробности автоматизации тестирования с использованием выбранного инструмента.

Шаги работы:

1) Анализ требований к приложению: оценить функциональные и нефункциональные требования к приложению, описанному в предыдущей работе.
2) Разработка плана тестирования: описать все необходимые элементы тестирования, включая идентификатор плана, вводную часть, тестируемые элементы, подход к тестированию, критерии прохождения и смоук-тестирования, график работ и другие.
3) Подготовка тестовой среды: настроить нужную среду для проведения тестов.
4) Автоматизация тестирования: выбрать инструмент для автоматизации тестирования (например, Selenium), создать скрипты для тестирования минимум трех функций приложения, провести тестирование.
5) Документирование результатов тестирования: зафиксировать все результаты выполнения автоматизированных тестов, включая скриншоты.
6) Анализ и предоставление рекомендаций: оценить результаты тестирования и предоставить рекомендации по улучшению приложения.

2 РАЗРАБОТКА ПЛАНА ТЕСТИРОВАНИЯ

2.1 Идентификатор тестового плана

План тестирования Шифратор и Дешифратор (TST-001)

2.2 Ссылки на используемые документы

1) Практическая работа №3 – Методологии TDD и BDD.
2) Стандарт IEEE Std 829-2019 для тестирования ПО.

2.3 Введение

Цель данного тестового плана — гарантировать, что шифратор и дешифратор работают корректно, соблюдают все требования, и что все функции полностью покрыты тестами.

2.4 Тестируемые элементы

1) Шифрование файла методом Цезаря.
2) Дешифрование файла методом Цезаря.
3) Шифрование файла методом XOR.
4) Дешифрование файла методом XOR.

2.5 Проблемы риска тестирования ПП

1) Ошибки в логике шифрования или дешифрования.
2) Потеря данных при шифровании/дешифровании.
3) Неверное использование ключей шифрования.
4) Нарушение структуры файла после шифрования.

2.6 Особенности или свойства, подлежащие тестированию

1) Правильность работы методов шифрования и дешифрования.
2) Сохранность данных после декодирования.
3) Проверка использования корректных ключей для шифрования/дешифрования.
4) Стабильность шифратора при больших объемах данных.

2.7 Особенности (свойства), не подлежащие тестированию

Графический интерфейс (если приложение консольное).

2.8 Подход

Для тестирования используются автоматизированные модульные тесты (TDD), а также сценарии на основе поведения (BDD), написанные на Python с использованием unittest и behave.

2.9 Критерии смоук-тестирования

1) Успешная инициализация шифратора.
2) Успешное шифрование и дешифрование файла.
3) Успешное завершение операции без ошибок.

2.10 Критерии прохождения тестов

Все тесты считаются пройденными, если:

1) Шифрование и дешифрование файлов работает корректно и сохраняет исходные данные.
2) Использование ключей шифрования не вызывает ошибок.
3) Шифрование и дешифрование завершаются успешно.

2.11 Критерии приостановки и возобновления работ

Приостановка тестирования при критических ошибках (например, невозможность дешифровки). Возобновление тестирования – после устранения ошибок.

2.12 Тестовая документация

Документация включает:

1) Настоящий тест-план.
2) Отчеты о прохождении тестов.

2.13 Основные задачи тестирования

1) Обеспечение корректного выполнения функций шифратора и дешифратора.
2) Выявление и исправление ошибок, влияющих на функциональность.

2.14 Необходимый персонал и обучение

Необходим один тестировщик, знакомый с Python и библиотеками unittest и behave.

2.15 Требования среды

1) Установленный интерпретатор Python.

2) IDE (например, PyCharm) или текстовый редактор для работы с кодом.
3) Среда для запуска модульных тестов и сценариев.

2.16 Распределение ответственности

Ответственность за написание и проведение тестов лежит на тестировщике.

2.17 График работ (календарный план)

1) Разработка плана тестирования — 1 день.
2) Написание и запуск тестов — 2 дня.
3) Исправление ошибок — 1 день.
4) Подготовка отчета — 1 день.

2.18 Риски и непредвиденные обстоятельства

1) Ошибки в коде, требующие значительного времени на исправление.
2) Изменение требований к шифратору.

2.19 Утверждение плана тестирования

План тестирования утверждается ответственным за тестирование и разработку.

2.20 Глоссарий

1) Тестируемый элемент: Функция или модуль, который проверяется тестами.
2) Позитивный тест: Тест, проверяющий корректную работу функции при корректных входных данных.
3) Негативный тест: Тест, проверяющий поведение функции при некорректных данных.

3 АВТОМАТИЗАЦИЯ ТЕСТИРОВАНИЯ WEB-Приложения

3.1 Тест 1

В расширении Selenium IDE был написан тестовый скрипт, который должен открывать поисковую строку google.com, вбивать в нее запрос “Selenium Web Driver”, нажимать на кнопку поиск и открывать первую ссылку в браузере

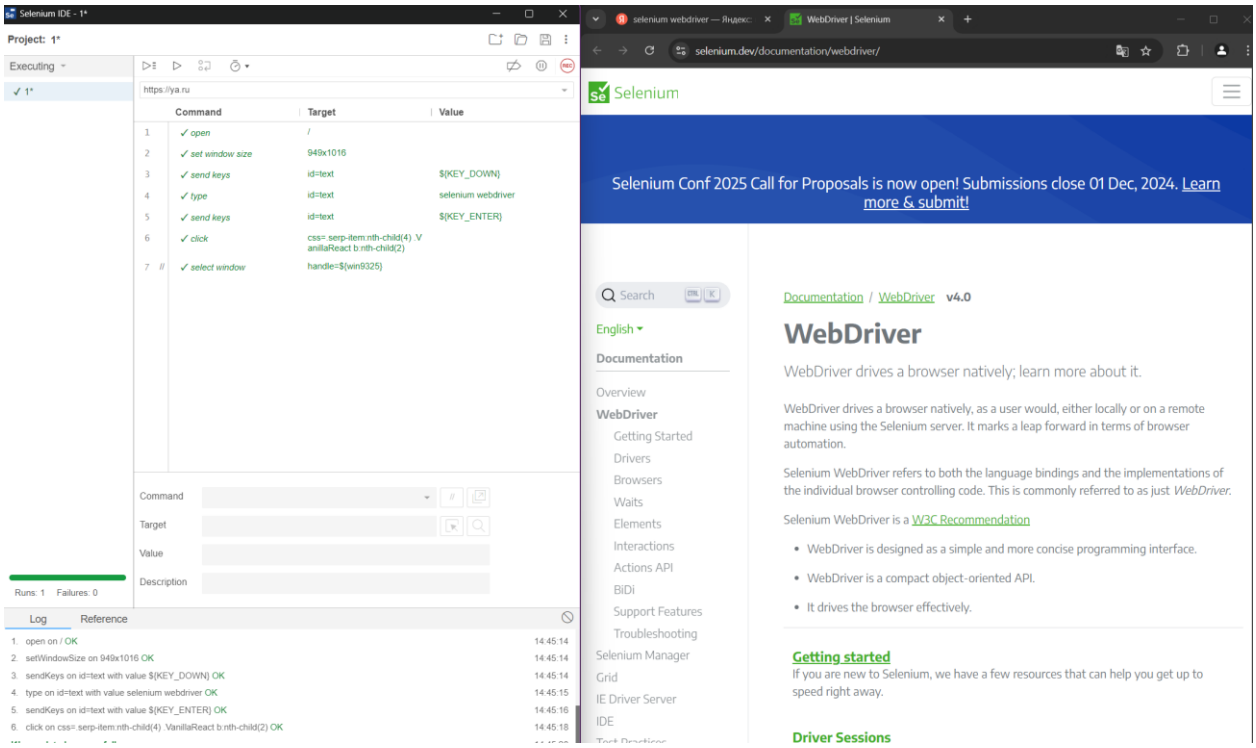

Рисунок 1. Тестовый скрипт 1

3.2 Тест 2

В расширении Selenium IDE был написан тестовый скрипт, который должен открывать статью Selenium в википедии и переходить в этой статье на статью о языке программирования JavaScript.

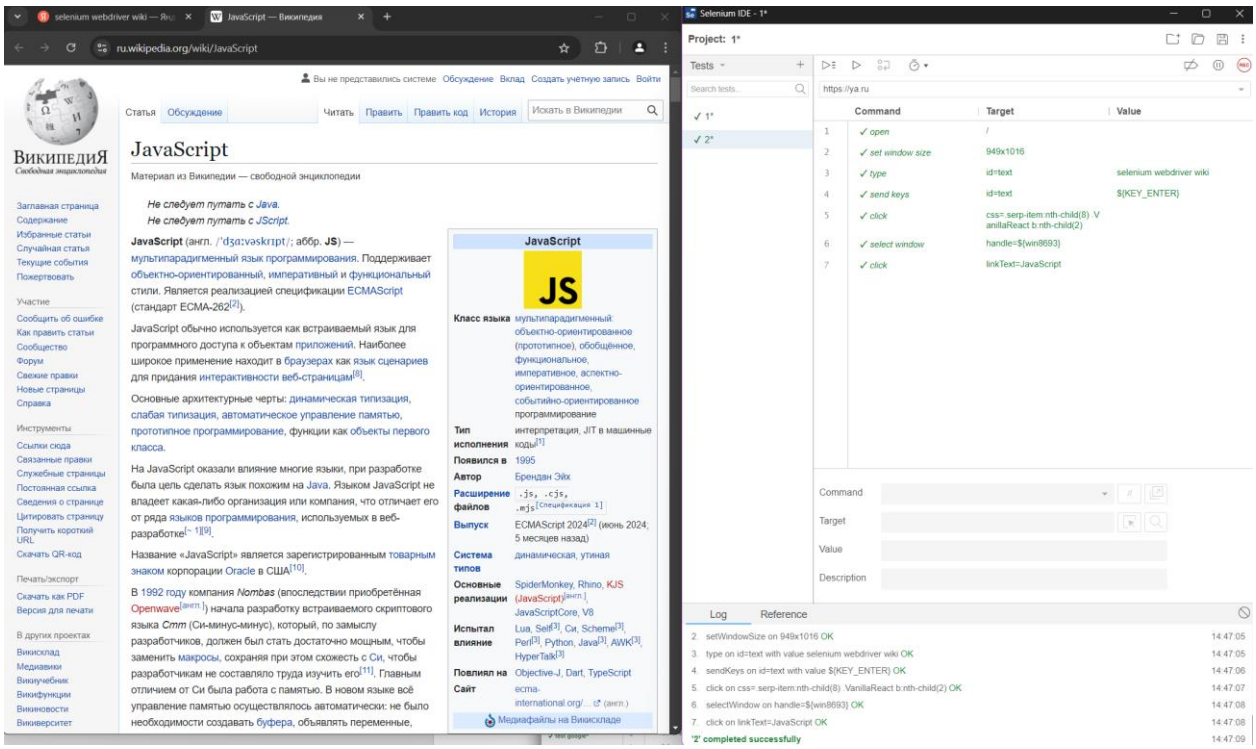

Рисунок 2. Тестовый скрипт 2

3.3 Тест 3

В расширении Selenium IDE был написан тестовый скрипт, который должен открывать регистрацию на сайте habr.ru, вводить данные во все поля, которые требует форма регистрации, принимать все пункты пользовательского соглашения и нажимать на кнопку подтверждения регистрации.

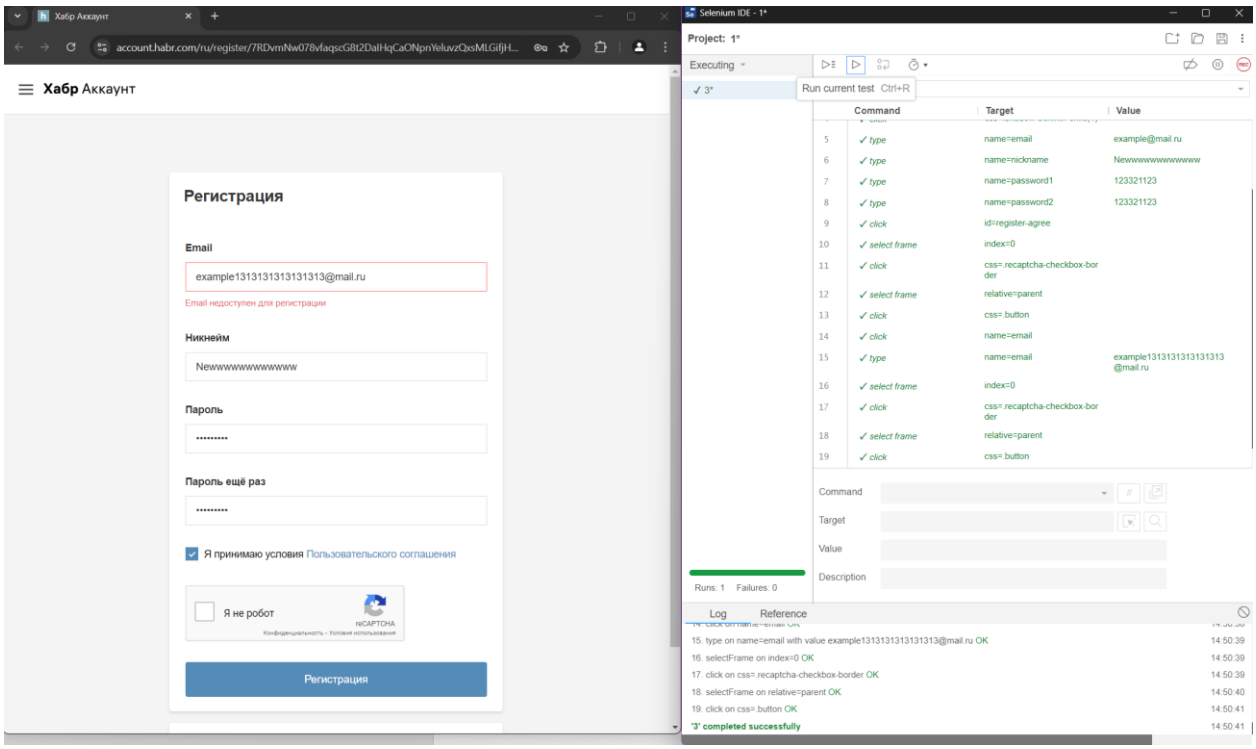

Рисунок 3. Тестовый скрипт 3

4 ВЫВОД

Все основные функции приложения для шифрования и дешифрования файлов, такие как шифрование методом Цезаря и XOR, работают корректно. В процессе тестирования были обнаружены и исправлены небольшие ошибки, что повысило надежность работы программы. Автоматизированные тесты на Python обеспечили полный охват функциональности, помогая выявить и устранить все недочеты. Приложение корректно обрабатывает файлы при шифровании и успешно восстанавливает исходные данные при дешифровке. Серьезных рисков в работе приложения не выявлено, но возможны доработки при изменении требований. Приложение готово к выпуску. Также освоены навыки проведения автоматизированного тестирования Web-приложений.