План тестирования

1. Введение

1.1. Основная информация

Документ описывает методы и подходы к тестированию проекта. Объект тестирования — это деятельность, направленная на проверку работоспособности функций проекта.

1.2. Цель

Тест-план проекта преследует следующие цели:

- Определить существующую информацию о проекте подлежащую тестированию.
- Описать стратегии тестирования, которые будут использоваться.
- Описать тестовые сценарии, которые будут использоваться для проверки функциональности и соответствия требованиям проекта.
- Определить необходимые ресурсы для проведения работ по тестированию.
- Привести результаты тестирования.

Результаты будут отправлены преподавателю в виде отчетов.

1.3. Описание продукта

Программа представляет собой визуализатор алгоритма Ахо-Корасик. Визуализатор алгоритма - программное средство, позволяющее наглядно отобразить процесс и результаты работы алгоритма.

Основная цель визуализатора - помочь пользователям лучше понять и визуально представить, как алгоритм Ахо-Корасик строит конечный автомат, которому затем передаёт строку поиска. Автомат получает по очереди все символы строки и переходит по соответствующим рёбрам. Если автомат пришёл в конечное состояние, соответствующая строка словаря присутствует в строке поиска.

2. Стратегия тестирования

2.1. Типы тестирования

Функциональное: цель функционального тестирования состоит в том, чтобы убедиться, что весь программный продукт работает в соответствии с требованиями, и в приложении не появляется существенных ошибок.

Нефункциональное: Нефункциональное тестирование описывает тесты, необходимые для определения характеристик программного обеспечения, измеряемые различными величинами:

- Нагрузочное тестирование
- Объёмное тестирование

2.2. Подход к тестированию

Подход к тестированию визуализатора алгоритма Ахо-Корасик следующий:

- Идентификация функциональных требований: определение основных функций и возможностей визуализатора, которые требуется протестировать.
- Разработка тестовых сценариев: создание набора тестовых сценариев, которые покрывают все основные функции и возможности визуализатора. Каждый сценарий должен содержать шаги для проверки конкретных требований.
- Разработка тестовых данных: создание тестовых данных, которые будут использоваться при выполнении тестовых сценариев. Это может включать создание вводных данных различной сложности, со случайными или заданными свойствами.

- Выполнение тестовых сценариев: запуск тестовых сценариев с использованием визуализатора и проверка результатов выполнения каждого шага и конечного результата
- Регистрация и анализ результатов: фиксация результатов выполнения каждого тестового сценария и анализ полученных данных. Если обнаружены ошибки, они должны быть документированы и переданы разработчикам.

2.3. Критерии завершения тестирования

Критерий завершения тестирования визуализатора алгоритма Ахо-Корасик определяется как достижение требуемого уровня качества и надежности, а также выполнение всех функциональных требований и ожиданий пользователей. Это может быть подтверждено успешным выполнением всех тестовых сценариев, отсутствием критических ошибок.

3. Ресурсы

3.1. Оборудование и программное обеспечение

Для тестирования визуализатора алгоритма Ахо-Корасик необходимо следующее оборудование и программное обеспечение:

- Компьютер или сервер с достаточными ресурсами для запуска визуализатора и обработки данных.
- Операционная система, совместимая с выбранным визуализатором.
- IntelliJ IDEA

Тестирование может проводиться как на реальном оборудовании и программном обеспечении, так и на виртуальных машинах.

Тестовые данные

Тестовые данные для визуализатора алгоритма Ахо-Корасик включают различные вводные тексты и наборы шаблонов (разделенные #).

Тестовые сценарии включают:

- Визуализацию процесса построения конечного автомата по алгоритму Ахо-Корасик.
- Визуализацию различных шагов алгоритма, например, выбор следующего ребра или вершины.

Тестовые данные могут быть созданы вручную. Важно убедиться, что визуализатор обрабатывает их корректно и не выдает неправильные результаты.

4. План тестирования

4.1. Тестирование основных функций продукта

- Тест на текст и один шаблон:
  - Входные данные: текст и шаблон.
  - Ожидаемый результат: алгоритм должен вернуть вхождения подстроки в текст и вывести визуализацию бора и конечного автомата.
- Тест на текст, в котором несколько шаблонов:
  - Входные данные: текст и набор шаблонов написание которых разделяется “#”.
  - Ожидаемый результат: алгоритм должен вернуть вывести визуализацию бора и конечного автомата.
- Тест на длинный текст вершин и малое количество шаблонов:
  - Входные данные: текст и шаблон.
  - Ожидаемый результат: алгоритм должен вернуть вхождения подстроки в текст и вывести визуализацию бора и конечного автомата.
- Тест на короткий текст вершин и большое количество шаблонов:
  - Входные данные: текст и шаблон.

    - Ожидаемый результат: алгоритм должен вернуть вхождения подстроки в текст и вывести визуализацию бора и конечного автомата.
- Тест на длинный текст и большое количесвто шаблонов:
    - Входные данные: текст и шаблон.
    - Ожидаемый результат: алгоритм должен вернуть вхождения подстроки в текст и вывести визуализацию бора и конечного автомата.
- **Тестирование граничных условий**
- Тест на пустой текст:
    - Входные данные: пустой текст.
    - Ожидаемый результат: вывод предупреждения на экране об отсутвии текста.
- Тест на отсутствие шаблонов:
    - Входные данные: непустой текст и 0 шаблонов.
    - Ожидаемый результат: вывод предупреждения об отсутствии шаблонов.
- Тест на повторяющиеся шаблоны:
    - Входные данные: текст и набор шаблонов написание которых разделяется “#” и при этом некоторые одинаковые.
    - Ожидаемый результат: алгоритм должен вернуть вхождения подстроки в текст исключив лишние шаблоны и вывести визуализацию бора и конечного автомата.

- Тест на то что один из шаблонов больше текста:
    - Входные данные: текст и набор шаблонов написание которых разделяется “#” и при один из них больше текста.
    - Ожидаемый результат: алгоритм должен вернуть вхождения подстроки в текст исключив большие шаблоны и вывести

визуализацию бора и конечного автомата без учёта этого шаблона.

4.2. Тестирование интерфейса

- Тестирование интерфейса:
  - Входные данные: текст и набор шаблонов написание которых разделяется “#”.
  - Ожидаемый результат: алгоритм должен вернуть вывести визуализацию бора и конечного автомата.
- Тест на случай отсутствия текста:
  - Входные данные: пустой текст.
  - Ожидаемый результат: вывод предупреждения на экране об отсутвии текста.
- Тест на случай некорректных входных данных:
  - Входные данные: некорректный текст и набор некорректных шаблонов написание которых разделяется “#”.
  - Ожидаемый результат: алгоритм должен обработать данную ситуацию и вернуть сообщение об ошибке или пустой результат.

- **Ручное тестирование**
  - Проверка навигации и функциональности: проход по всем разделам и функциям интерфейса, чтобы убедиться, что они работают правильно.
  - Проверка отображения элементов: убедиться, что все элементы интерфейса правильно отображаются (тексты, изображения, кнопки и т. д.).
  - Проверка взаимодействия: протестировать взаимодействие с интерфейсом (нажимать кнопки, заполнять формы, выполнять действия) и убедиться, что все работает ожидаемым образом.

4.3. Тестирование структуры данных

- Тест на добавление вершины в граф:
  - Входные данные: пустой граф, команда добавления вершины.

- Ожидаемый результат: вершина успешно добавлена в граф.

– Тест на удаление вершины из графа:
  - Входные данные: граф с несколькими вершинами, команда удаления одной из вершин.
  - Ожидаемый результат: вершина успешно удалена из графа и связанные с ней ребра также удалены.

– Тест автомата на набор с большим количеством похожих, но не равных шаблонов:
  Входные данные: граф с несколькими вершинами, команда удаления одной из вершин.
  - Ожидаемый результат: вывод графа и бора с корректными терминальными суффиксами и ссылками.

– Тест на удаление терминальных ссылок при смене терминального состояния:
  - Входные данные: вершина , команда смена терминального состояния.
  - Ожидаемый результат: конечный автомат перестроен, в боре цвет поменялся, добавился\удалился шаблон.

– Тест на бор и конечный автомат из одного шаблона:
  - Входные данные: текст и шаблоны
  - Ожидаемый результат: конечный автомат и бор представляют собой однонаправленный список с единственным терминальным состоянием в конце.

– Тест на бор и конечный автомат из шаблонов, которые являются префиксом другого шаблона:
  - Входные данные: текст и шаблоны, каждый шаблон является подмножеством другого шаблона.
  - Ожидаемый результат: конечный автомат и бор представляют собой однонаправленный список с единственным терминальным состоянием в разных местах.

5. Риски и ограничения

5.1. Идентификация потенциальных рисков

Потенциальные риски при тестировании алгоритма Ахо-Корасик:

- Неправильная реализация алгоритма.
- Недостаточное покрытие тестами: Если не все возможные сценарии использования алгоритма покрыты тестами, то могут быть упущены ошибки или проблемы в его работе.
- Неправильный выбор тестовых данных: Если тестовые данные не являются репрезентативными для реальных сценариев использования алгоритма Ахо-Корасик, то результаты тестирования могут быть искажены.
- Проблемы с производительностью: Если алгоритм работает слишком медленно или требует слишком много ресурсов, то это может быть проблемой. Нужно оценить производительность алгоритма и убедиться, что он работает достаточно быстро и эффективно.
- Неправильная интерпретация результатов: Если результаты тестирования неправильно интерпретируются или анализируются, то могут быть сделаны неверные выводы о работе алгоритма.

5.2. Описание ограничений тестирования

Ограничения тестирования алгоритма Ахо-Корасик могут включать:

- Ограниченные ресурсы: Тестирование может быть ограничено доступными ресурсами, такими как вычислительная мощность или память. Это может ограничить возможность проведения тестов на больших наборах данных.

- Ограниченное время: Время, выделенное для тестирования, может быть ограничено. Это может ограничить количество и сложность тестов.
- Ограничения на входные данные: Алгоритм Ахо-Корасик может иметь ограничения на входные данные, например, размер текста или шаблонов. Тестирование должно учитывать эти ограничения и проверять, как алгоритм обрабатывает граничные случаи.
- Ограничения на выходные данные: Алгоритм Ахо-Корасик может иметь ограничения на выходные данные, например, максимальный размер Конечного Автомата. Тестирование должно учитывать эти ограничения.

6. Расписание тестирования

Планируемые даты окончания каждого типа тестирования:

10 июля (воскресенье) – тестирование основных функций продукта.

11 июля (вторник) – тестирование структуры данных.

13 июля (четверг) – тестирование интерфейса, граничных условий.