Федеральное государственное автономное образовательное учреждение высшего образования "Национальный Исследовательский Университет ИТМО"
Мегафакультет Компьютерных Технологий и Управления
Факультет Программной Инженерии и Компьютерной Техники

Вариант №133205
Лабораторная работа 1

по дисциплине

Тестирование программного обеспечения

Выполнил Студент группы P33102
Лапин Алексей Александрович
Преподаватель:
Харитонова Анастасия Евгеньевна

г. Санкт-Петербург
2024г.

Цели работы:

1. Для указанной функции провести модульное тестирование разложения функции в степенной ряд. Выбрать достаточное тестовое покрытие.

2. Провести модульное тестирование указанного алгоритма. Для этого выбрать характерные точки внутри алгоритма, и для предложенных самостоятельно наборов исходных данных записать последовательность попадания в характерные точки. Сравнить последовательность попадания с эталонной.

3. Сформировать доменную модель для заданного текста. Разработать тестовое покрытие для данной доменной модели

Вариант:

1. Функция $\sin(x)$

2. Программный модуль для обхода неориентированного графа методом поиска в глубину (http://www.cs.usfca.edu/ galles/visualization/DFS.html)

3. Описание предметной области:

Форд отказался от попыток уснуть. В углу его каюты стоял маленький компьютер. Он посидел за ним немного, пытаясь сочинить новую статью о вогонах для "Путеводителя но не смог выдумать ничего достаточно едкого и бросил. Он надел халат и решил сходить на мостик.

Обоснование выбора тестируемых значений

Фунция

Определяем области эквивалентности для аргумента функции:

1. Числа от [-1e10; 1e10]

2. NaN, бесконечность

3. {x : x < -1e10 || x > 1e10}

Области эквивалентности для eps:

1. $[EPS\_LIMITS; 1]$

2. eps : eps < EPS_LIMITS

Берем несколько значений из каждой области эквивалентности и делаем тесты.

Алгоритм

Сценарии использования:

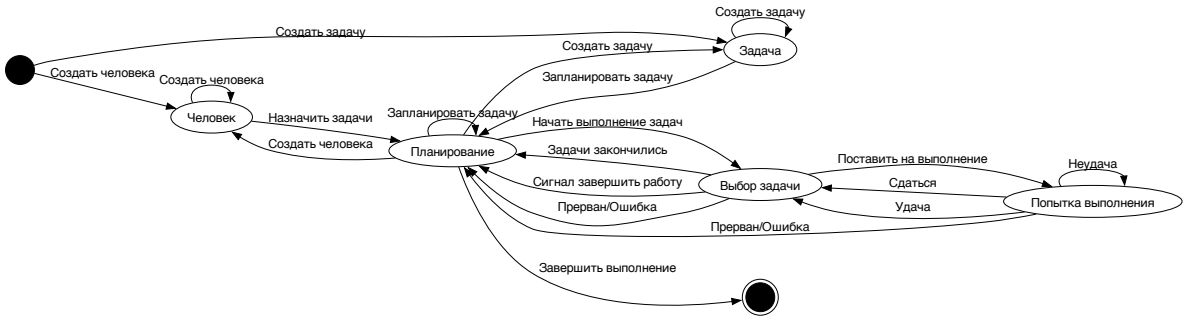
Прецедент: LoadGraphAndGetCorrectTraversal	
ID:	1
Краткое описание:	Пользователь создает граф, добавляя в него ребра, пользователь задает стартовую вершину, программа делает корректный обход
Главные актёры:	Пользователь
Основной поток:	Пользователь создает граф, добавляя в него ребра, пользователь задает стартовую вершину, программа делает корректный обход

Прецедент: AddVertexThatExistsThrowingIllegalArgumentException	
ID:	2
Краткое описание:	Пользователь добавляет вершину, которая уже существует. Программа выдает ошибку.
Главные актёры:	Пользователь
Основной поток:	1. Пользователь создает граф. 2. Пользователь добавляет вершину 3. Пользователь добавляет ту же вершину снова 4. Программа выдает IllegalArgumentException

Прецедент:	TraverseStartsOnVertexThatDoesntExistThrowingIllegalArgumentException
ID	3
Краткое описание:	Пользователь начинает обход графа с несуществующей вершины
Главные актёры:	Пользователь
Основной поток:	1. Пользователь создает граф 2. Пользователь добавляет вершины 3. Пользователь передает в стартовую вершину обхода, вершину которой нет в графе. 4. Пользователь получает IllegalArgumentException

Берем несколько значений и проверяем для каждого сценария использования.

Доменная модель



Покроем неэквивалентные пути тестами. Также сделаем тесты на ошибки и прерывания.

Описание реализации доменной модели

У нас есть возможность создавать людей, назначать им задания и требовать, чтобы они их выполнили. Эти задания выполняются в отдельном потоке, позволяя основному потоку продолжать работу. Каждое задание имеет вероятность успеха и будет выполняться до тех пор, пока не будет либо успешно завершено, либо брошена. Кроме того, у каждого человека есть ограничение на количество задач, которые он может выполнять одновременно. Люди будут продолжать работать до тех пор, пока все поставленные перед ними задачи не будут выполнены или брошены. Наконец, мы можем получить количество задач, которые человек успешно выполнил за время работы.

1 Вывод

В этой лабораторной работе я познакомился с модульным тестированием функций, алгоритмов и доменных моделей.

Для функции $\sin(x)$ мы разбили области значений аргумента и точности eps на эквивалентные классы. Затем выбрали несколько тестовых значений из каждого класса для проверки корректности разложения функции в степенной ряд.

Для алгоритма обхода графа в глубину были рассмотрены основные сценарии использования:

1. Корректный обход при правильных входных данных
2. Обработка ошибки при добавлении уже существующей вершины
3. Обработка ошибки при попытке начать обход с несуществующей стартовой вершины

Для каждого сценария были подобраны соответствующие тестовые наборы данных.

Для доменной модели, описывающей систему назначения и выполнения заданий людьми, была построена диаграмма состояний. На ее основе были выделены пути, которые необходимо покрыть тестами.

В результате выполнения работы я получил практические навыки составления тестовых наборов для проверки функций, алгоритмов и доменных моделей. Я научился применять такие методы как разбиение на классы эквивалентности, анализ сценариев использования, построение диаграмм состояний. Познакомился с JUnit5, научился использовать параметризованные тесты.

Это позволит мне в дальнейшем эффективно заниматься тестированием разрабатываемого программного обеспечения.