Университет ИТМО
Факультет программной инженерии и компьютерной техники

Лабораторная работа №1

по дисциплине «Тестирование программного обеспечения»

Выполнил:

Студент группы P3331

Дворкин Борис Александрович

Вариант: 31004

Преподаватель:
Гаврилов Антон Валерьевич

г. Санкт-Петербург

2024 г.

Содержание

Описание задания

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 31004

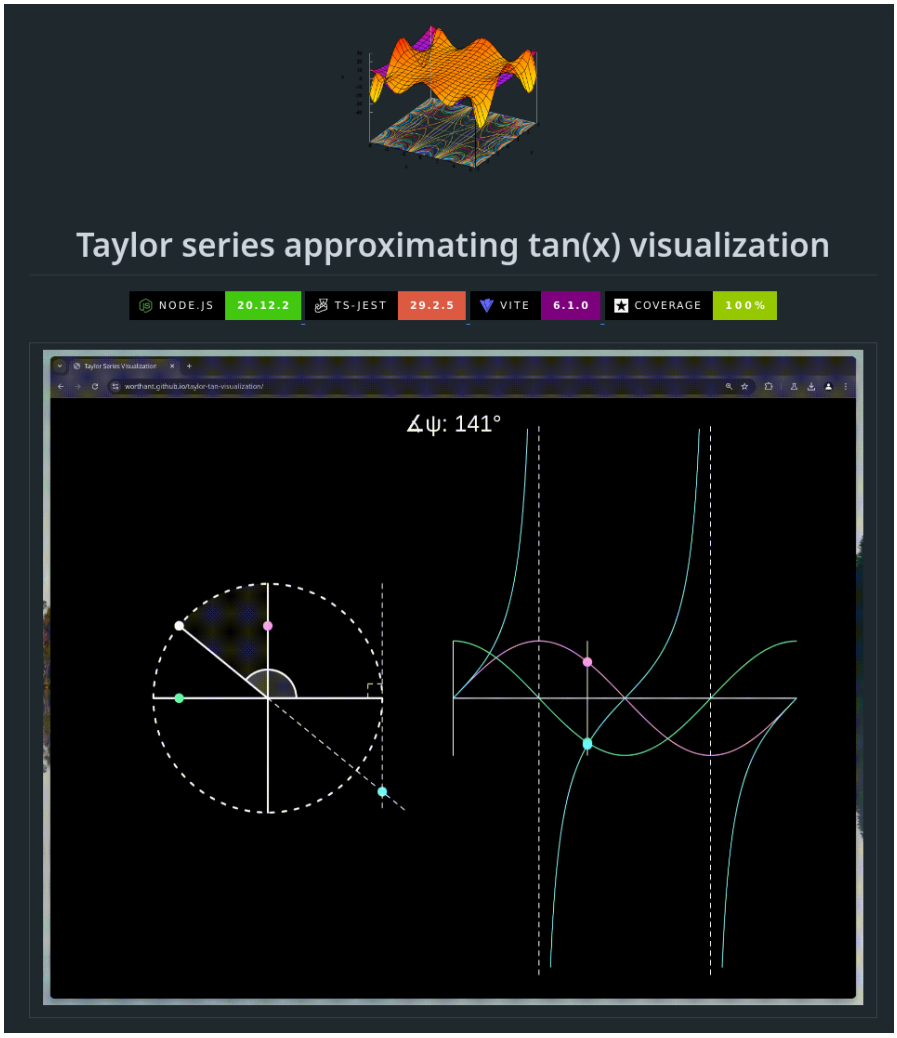
1. Функция tg(x)
2. Программный модуль для обхода неориентированного графа методом поиска в глубину (http://www.cs.usfca.edu/~galles/visualization/DFS.html)
3. Описание предметной области:

Много-много миллионов лет назад раса гиперразумных всемерных существ (чье физическое проявление в их всемерной вселенной практически не отличается от нашего) так устала от постоянных споров о смысле жизни, которые отвлекали их от их излюбленного времяпрепровождения -- брокианского ультра-крикета (забавная игра, заключающаяся в том, чтобы неожиданно ударить человека без видимой на то причины и убежать) -- что решила сесть и решить все вопросы раз и навсегда.

Модульное тестирование tg(x)

Мне было скучно, поэтому первым делом я решил реализовать функцию тангенса вручную, не используя библиотеку Math, с помощью какого-то степенного ряда (как и сказано в ТЗ, в целом). Сделал я это с помощью ряда Тейлора (Маклорена), ибо он лучше ведёт себя около нуля, нежели ряд Фурье, и также его полиномы просты в реализации.

Затем, так как это всё ненаглядно - я нашёл классную библиотеку p5.js и сделал мощную визуализацию в стиле 3dBlueBrown.



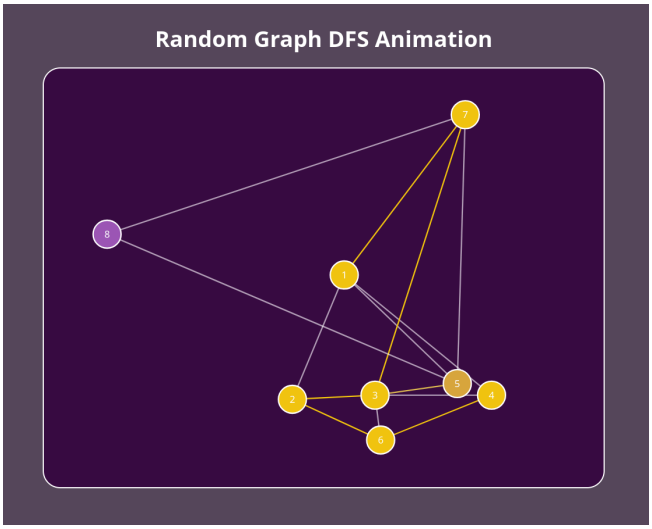
Таким образом, я создал кодовую базу с различными функциями для тестирования, для модульного взял jest (ts-jest), а для pbt взял fast-check. Проверил свойства монотонности, область значений функций, чётность/нечётность, бесконечные аргументы и т.д.

Подробнее можно посмотреть тут:

Репа: https://github.com/worthant/taylor-tan-visualization
Визуализация: https://worthant.github.io/taylor-tan-visualization/

Тестирование белым ящиком DFS

Пока энтузиазм не утих, поэтому реализовав DFS на typescript, а затем на javascript двумя способами - рекурсивным и со стэком, я тоже захотел как-то это визуализировать. Так как с этой части лабораторной я начинал, то мне было просто жесть как скучно, и я сделал визуализацию на чистом html, js, css + canvas, без каких-либо логических прослоек.

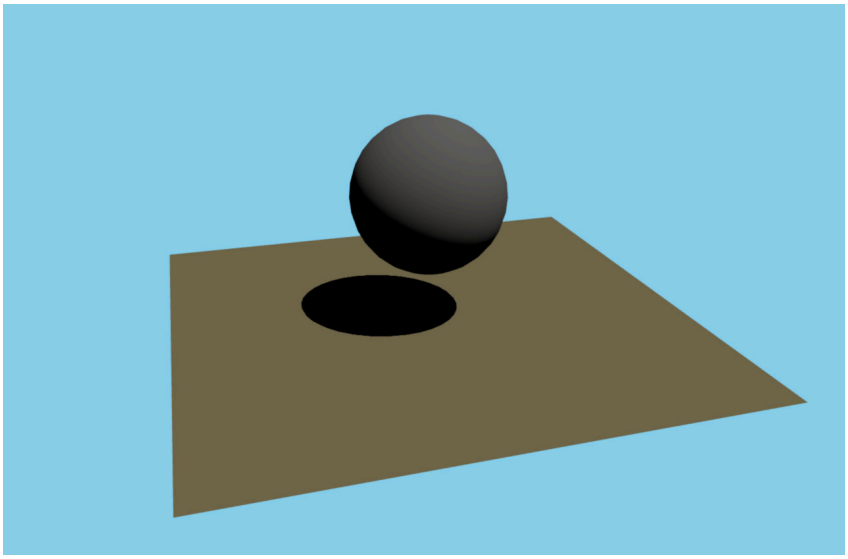


Опять же, создав себе простор для тестирования, запилил unit и pbt тесты с помощью тех же тулов что и для тангенса. Тестирование делал белым ящиком, проверял поведение DFS на различных деревьях - сохранение правильного порядка, отсутствие дубликатов после прохождения по узлам и т.д.

Подробнее можно посмотреть тут:

Репа: https://github.com/Imtjl/dfs-2d
Визуализация: https://imtjl.github.io/dfs-2d/

P.S.: Изначально амбиции были сделать dfs в 3d, и я честно сел, создал плоскость, свет, камеру, тени, уже начал реализовывать основную логику, но понял что не успеваю по времени, в виду наличия огромного количества предметов, и работы 🙂



Реализация доменной модели

Считаю что эта часть лабы - то, за что должно быть стыдно ВТ, ровно как и аналогичное порождение этого задания в курсе проги в первом семестре.

Ну хоть потыкал ООП на тайпскрипте, было интересно, и очень просто. Создал простейший класс HyperRace для рассы сверхразумов, дал им уровни споров, насроение для игры в УЛЬТРА КРИКЕТ и состояние решённых проблем (вот бы мне так). Затем научил играть в крикет, исправлять свои проблемы, сбрасывать споры, ну и создал метод для получения состояния класса (на жавовском, сделал геттер для получения приватных полей, ура инкапсулирование). На доп сделал ещё один класс HyperTeam, где научил оболтусов спорить не всей рассой, а конкретной командой от 0 до 20 человек.

Тесты комментировать не хочется, ну написал тут инвариантов белого ящика, методы повызывал, ошибки поотлавливал.

Подробнее можно посмотреть тут:

Репа: https://github.com/lmtjl/qa-fundamentals/tree/main/unit-testing/domain-model

Вывод

Реализуя тестирование функции **tg(x)**, я не просто проверил корректность разложения в ряд Тейлора, но и создал полноценную **визуализацию тригонометрических функций**. Визуализация являлась неким способом приемочного тестирования, так я убедился в корректности расчетов и функционировании написанной математики, и глубже понял их геометрический смысл. Кроме того, работа с тестированием математических функций привела к изучению свойств численных вычислений в языке, таких как **накопление ошибок при суммировании ряда**, влияние **погрешности округления**, устройство системы типов и точность вычислений в JavaScript. Это вообще отдельная головная боль, ибо вычисления в этом языке ужасны, но писать на нём (тут щас меня осудят) во многом приятно ввиду сравнительной простоты синтаксиса и удобной экосистемы, поэтому все возникающие мелкие проблемы нивелируются удобством. Помимо этого попробовал *мутационно* протестировать модули, вручную внося изменение в поведение функций, и смотря отловил ли эти изменения тесты, таким образом удалось выявить, что я составил хорошее тестовое покрытие, ибо все таким ошибки мои тесты поймали.

Касательно DFS аналогично, было очень круто после всех курсов алгоритмов, функционального программирования и прочих, с новыми силами ворваться и с лёгкостью реализовать алгоритмы прохода по графам, сравниваешь себя текущего с версией года два назад и видишь насколько сильно растешь в первую очередь как инженер-разработчик, решая казалось бы не самые тривиальные задачи, за которые раньше было бы страшно даже взяться. Тестирование ничего конкретного не выявило, потому что ошибиться в этом алгоритме, как будто бы, сложно, но практика позволила посмотреть под другим углом на теорию графов, и заняться изучением компьютерной графики. Таким образом, превращаем нудную лабораторную в интересное времяпрепровождение 🙂.

Работа с доменной моделью раскрыла ООП в ts изнутри. Вспомнил про базовые концепции инкапсуляции, написал тривиальнейшие методы, даже практически отвращения не испытал.

Пригодится оно или нет, благодаря моему энтузиазму выполнение лабы не уткнулось в прямое выполнение наискучнейших заданий, а послужило увлекательным занятием по применению областей вышмата, теории графов, алгосов и структур данных и веб разработки - это значительно расширило границы моих знаний, и дало базу в изучении концепций компьютерной графики, а также я законтрибутил в либу p5.js и возможно поеду по приколу к ним на стажку в Германию, когда буду там учиться xd

P.S.: Репа где части лабы лежат сабмодулями и задокументированы вот: https://github.com/lmtjl/qa-fundamentals/tree/main/unit-testing