

Университет ИТМО

**Тестирование программного обеспечения**

**Лабораторная работа №1**

Вариант 681

Выполнила: Калугина Марина

Группа: Р3402

г. Санкт-Петербург

2020 г.

# Задание

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

1. Функция  $\arcsin(x)$
2. Программный модуль для работы с AVL-деревом (<http://www.cs.usfca.edu/~galles/visualization/AVLtree.html>)
3. Описание предметной области:

Англия больше не существует. Каким-то образом он смог это осознать. Он попытался еще раз. Америки больше нет, подумал он. Это не охватывалось. Нужно снова попробовать что-нибудь помельче. Нью-Йорка нет. Не получается. Он все равно никогда всерьез не верил, что он существует. Доллар упал навсегда. Что-то ощущается. Все фильмы с Богартом пропали, сказал он себе, и ему стало тоскливо. Макдональдс, вспомнил он. Больше никогда не будет маковских гамбургеров.

## Описание используемого JUnit

Используется JUnit5 из `org.junit.jupiter.api.*`

JUnit - библиотека для модульного тестирования в Java.

В JUnit Jupiter имеются следующие артефакты:

- `junit-jupiter-api` — JUnit Jupiter API для написания тестов и расширений;
- `junit-jupiter-engine` — реализация тестовой среды JUnit Jupiter, которая необходима только во время выполнения;
- `junit-jupiter-params` обеспечивает поддержку для параметризованных тестов в JUnit Jupiter;
- `junit-jupiter-migrationsupport` обеспечивает поддержку миграции с JUnit 4 на JUnit Jupiter и требуется только для запуска выбранных правил JUnit 4.

## **Используемые аннотации в лабораторной работе:**

@Test - аннотация для объявления тестов

@BeforeEach - аннотация для выполнения метода перед каждым тестом

@AfterEach - аннотация для выполнения метода после каждого теста

## **Прочие аннотации:**

@BeforeAll/@AfterAll - аннотация для выполнения метода перед каждым классом

@ParameterizedTest - аннотация для обозначения того, что метод является параметризованным тестом

@Tag - аннотация для пометки тестов для возможности фильтрации

@Disabled - аннотация для отключения тестового сценария

@RepeatedTest - аннотация для обозначения того, что метод является шаблоном для повторного теста. Такие методы наследуются, если они не переопределены.

@TestFactory - аннотация означает, что метод является тестовой фабрикой для динамических тестов.

@Nested - аннотация означает, что аннотированный класс является нестатическим вложенным тестовым классом.

@Timeout - используется для определения тайм-аута для метода или всех тестируемых методов

@TestTemplate - означает, что метод является шаблоном для тестовых случаев

@TestMethodOrder - используется для настройки порядка выполнения тестового метода

@TestInstance - используется для настройки жизненного цикла тестового экземпляра

@DisplayName - настраиваемое отображаемое имя для тестового класса или тестового метода.

@DisplayNameGeneration - объявляет настраиваемый генератор отображаемых имен для тестового класса.

# Исходный код

<https://github.com/KaluginaMarina/FourthYearOfItmo/tree/master/testing/lab1>

## Вывод

1. Было проведено модульное тестирование функции разложения арксинуса в степенной ряд. Для тестирования использовался анализ эквивалентности: функция, в области допустимых значений была разбита на классы эквивалентности. По паре значений из каждой группы эквивалентных по своему влиянию на систему значений были протестированы, кроме того были отдельно протестированы граничные значения и значения, выходящие за область допустимых значений.

В ходе выполнения первого этапа лабораторной работы были получены навыки модульного тестирования при помощи анализа эквивалентности. Были разобраны основные принципы данного метода и применены на практике.

2. Был протестирован программный модуль для работы с AVL-деревом. Были выбраны характерные точки алгоритма и исходные наборы данных и сравнены с эталонной. В рамках характерных точек были выбраны следующие типы сценариев: различное добавление и удаление вершин: как в пустые деревья, так и в заполненные, удаление несуществующих элементов, проверка функций балансировки для левого и правого поворота для вершин с детьми и листьев дерева, и проверка корректного расчета высоты дерева и др.

В ходе выполнения второго этапа лабораторной работы были получены навыки выделения характерных точек в алгоритме для тестирования и поиск нетривиальных входных данных для покрытия крайних точек алгоритма.

3. Для тестирования доменной модели было разработано тестовое покрытие, которое покрывает всю функциональность доменной модели. Главным принципом написания тестов служило положение о том, что не должно существовать такой функциональной части, которая не была бы покрыта тестами.

В ходе выполнения данного этапа лабораторной работы были получены навыки выбора и описания таких сценариев, чтобы они покрывали всю функциональность рабочей программы

В целом в ходе выполнения лабораторной работы были получены навыки модульного тестирования программного обеспечения: функций и доменных моделей при помощи библиотеки JUnit.