

Лабораторная работа №7

Иванова Виктория 6203-010302D

Задание 1

Сделала так, чтобы все объекты типа `TabulatedFunction` можно было использовать в качестве объекта-агрегата в цикле `for-each`, извлекаемые объекты при этом имеют тип `FunctionPoint`. Интерфейс `TabulatedFunction` был расширен для реализации `Iterable<FunctionPoint>`. В классах `ArrayTabulatedFunction` и `LinkedListTabulatedFunction` были добавлены анонимные классы итераторов. В `ArrayTabulatedFunction` итератор использует индекс для доступа к элементам массива, в `LinkedListTabulatedFunction` итератор проходит по узлам связного списка. Метод `next()` был реализован с проверкой наличия следующих элементов и выбрасывает `NoSuchElementException` при попытке получить элемент за пределами. Метод `remove()` выбрасывает `UnsupportedOperationException`. Итераторы возвращают копии точек функций, а не оригинальные объекты.

Задание 2

Создала интерфейс `TabulatedFunctionFactory` с 3 перегруженными методами `createTabulatedFunction`, соответствующими конструкторам классов табулированных функций. Реализовала два класса-фабрики `ArrayTabulatedFunctionFactory` и `LinkedListTabulatedFunctionFactory` как вложенные публичные классы внутри соответствующих классов функций. В классе `TabulatedFunctions` было добавлено приватное статическое поле `factory` типа `TabulatedFunctionFactory`, инициализированное объектом `ArrayTabulatedFunctionFactory`. Реализовала метод `setTabulatedFunctionFactory`. Все существующие методы создания табулированных функций в классе `TabulatedFunctions` были модифицированы для использования фабрики вместо прямого вызова конструкторов.

Задание 3

В классе `TabulatedFunctions` добавила ещё три перегруженных версии метода `createTabulatedFunction()`, принимающие параметр `Class<? extends TabulatedFunction> functionClass`. Использовала механизм рефлексии для поиска конструкторов с помощью `getConstructor()` и создания объектов через `newInstance()`. Все исключения рефлексии `NoSuchMethodException`, `InstantiationException`, `IllegalAccessException`, `InvocationTargetException` перехватываются и преобразуются в `IllegalArgumentException` с сохранением исходного исключения в качестве причины. Еще реализовала метод `tabulate` с поддержкой рефлексии.

```
"C:\Program Files\Java\jdk-25\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.2.3\lib\idea_rt.jar=63900" -Dfile.encoding=UTF-8 -Dsur

Тестирование итераторов
ArrayTabulatedFunction:
(0.0, 2.0)
(3.75, 4.0)
(7.5, 7.0)
(11.25, 14.0)
(15.0, 22.0)
LinkedListTabulatedFunction:
(0.0, 2.0)
(3.75, 4.0)
(7.5, 7.0)
(11.25, 14.0)
(15.0, 22.0)
фабрики
class functions.ArrayTabulatedFunction
class functions.LinkedListTabulatedFunction
class functions.ArrayTabulatedFunction
рефлексия
class functions.ArrayTabulatedFunction
{(0.0, 0.0), (5.0, 0.0), (10.0, 0.0)}
class functions.ArrayTabulatedFunction
{(0.0, 0.0), (10.0, 10.0)}
class functions.LinkedListTabulatedFunction
{(0.0, 0.0), (10.0, 10.0)}
class functions.LinkedListTabulatedFunction
{(0.0, 0.0), (0.3141592653589793, 0.3090169943749474), (0.6283185307179586, 0.5877852522924731), (0.9424777960769379, 0.8090169943749474), (1.2566370614359172, 0.9510565162951535), (1.5707963267948966, 1.0)}
Process finished with exit code 0
```