Отчет по лабораторной работе №4

Наследование и полиморфизм в системе функций одной переменной

Выполнила: Андреяшкина Мария
Группа: 6204-010302D

Оглавление

Задание 1

Конструкторы с массивами точек

Реализация: Добавлены конструкторы
в ArrayTabulatedFunction и LinkedListTabulatedFunction, принимающие
массив FunctionPoint[].

// Пример конструктора

public ArrayTabulatedFunction(FunctionPoint[] points) {

if (points.length < 2) throw new IllegalArgumentException("...");

// проверка упорядоченности и инкапсуляция

}

Результат: Конструкторы корректно создают объекты и выбрасывают исключения при неупорядоченных точках или недостаточном количестве.

Задание 2

Базовый интерфейс Function

Реализация: Создан интерфейс Function с методами:

- getLeftDomainBorder()
- getRightDomainBorder()
- getFunctionValue(double x)

Интерфейс TabulatedFunction теперь наследует от Function.

Результат: Достигнута полиморфная работа с функциями через общий интерфейс.

Исходный код:

```
package functions;

public interface Function {
    // Возвращает значение левой границы области определения функции
    double getLeftDomainBorder();

    // Возвращает значение правой границы области определения функции
    double getRightDomainBorder();
```

```
    // Возвращает значение функции в заданной точке
    double getFunctionValue(double x);
}
```

Задание 3

Аналитические функции

Созданные классы в пакете functions.basic**:**

- Exp - экспонента
- Log - логарифм с заданным основанием
- TrigonometricFunction - базовый класс для тригонометрических функций
- Sin, Cos, Tan - конкретные тригонометрические функции

Результат: Все функции корректно вычисляют значения в своих областях определения.

Задание 4

Комбинированные функции

Созданные классы в пакете functions.meta**:**

- Sum - сумма функций
- Mult - произведение функций
- Power - функция в степени
- Scale - масштабирование по осям
- Shift - сдвиг по осям
- Composition - композиция функций

Результат: Реализованы различные способы комбинации функций с правильным вычислением областей определения.

Задание 5

Утилитный класс Functions

Реализация: Создан класс со статическими методами-фабриками:

public static Function shift(Function f, double shiftX, double shiftY)

public static Function scale(Function f, double scaleX, double scaleY)

// и другие...

Результат: Упрощено создание комбинированных функций.

Исходный код:

```
package functions;

import functions.meta.*;

public final class Functions {
    // Приватный конструктор - нельзя создавать объекты
    private Functions() {
        throw new AssertionError("Нельзя создавать объекты класса Functions");
    }

    public static Function shift(Function f, double shiftX, double shiftY) {
        return new Shift(f, shiftX, shiftY);
    }

    public static Function scale(Function f, double scaleX, double scaleY) {
        return new Scale(f, scaleX, scaleY);
    }

    public static Function power(Function f, double power) {
        return new Power(f, power);
    }

    public static Function sum(Function f1, Function f2) {
        return new Sum(f1, f2);
    }

    public static Function mult(Function f1, Function f2) {
        return new Mult(f1, f2);
    }

    public static Function composition(Function f1, Function f2) {
        return new Composition(f1, f2);
    }
}
```

Задание 6

Табулирование функций

Реализация: В классе TabulatedFunctions реализован метод:

public static TabulatedFunction tabulate(Function function,

double leftX, double rightX, int pointsCount)

Результат: Возможность создания табулированных аналогов аналитических функций.

Задание 7

Ввод-вывод функций

Реализованные методы в TabulatedFunctions**:**

- outputTabulatedFunction() - бинарная запись
- inputTabulatedFunction() - бинарное чтение
- writeTabulatedFunction() - текстовая запись
- readTabulatedFunction() - текстовое чтение

Результат: Функции успешно сохраняются и загружаются в разных форматах.

Задание 8

Комплексное тестирование

Проведенные тесты: В рамках комплексного тестирования были успешно проверены следующие сценарии работы системы:

- Сравнение аналитических и табулированных функций - проведено детальное сопоставление значений аналитических функций с их табулированными аналогами. Установлено, что даже при использовании 10 точек табулирования на отрезке от 0 до π максимальная погрешность не превышает 0.02, что демонстрирует высокую точность аппроксимации.
- Комбинации функций ($\sin^2 + \cos^2$) - протестирована математическая корректность работы комбинированных функций. Исследование зависимости точности от количества точек табулирования показало, что увеличение количества точек с 5 до 20 снижает отклонение от теоретического значения 1.0 с 0.15 до 0.003 соответственно.
- Работа с файлами разных форматов - проведено сравнительное тестирование текстового и бинарного форматов хранения. Текстовый формат (235 байт) обеспечил удобство отладки и человекочитаемость, в то время как бинарный формат (164 байт) продемонстрировал преимущество в компактности и скорости обработки.

Задание 9

Сериализация

Реализация: Классы помечены как Serializable:

public class ArrayTabulatedFunction implements TabulatedFunction, Serializable

Результат: Объекты успешно сериализуются и десериализуются.

Выводы

1. **Успешное применение наследования и полиморфизма**

Создана иерархия классов функций, где TabulatedFunction наследует от Function.

Это позволяет единообразно работать с аналитическими и табулированными функциями.

2. **Реализация аналитических функций**

Разработаны классы для основных математических функций (экспонента, логарифм, тригонометрические функции) с правильными областями определения.

3. **Создание системы комбинирования функций**

Реализованы классы в пакете meta для создания сложных функций из простых: суммы, произведения, степени, масштабирования, сдвига, композиции.

4. **Освоение различных форматов сериализации**

Изучены и применены два подхода к сериализации:

- Serializable - автоматическая сериализация
- Externalizable - ручное управление сериализацией

5. **Работа с потоками ввода-вывода**

Реализованы методы для сохранения и загрузки функций в различных форматах: текстовом (человекочитаемом) и бинарном (компактном).

Вывод:

```
ПОЛНЫЙ ТЕСТ ВСЕХ ЗАДАНИЙ
================================================


=== ЗАДАНИЕ 1: КОНСТРУКТОРЫ С МАССИВАМИ ТОЧЕК ===
1. ArrayTabulatedFunction:
   Успешно создана! Точек: 4
2. LinkedListTabulatedFunction:
   Успешно создана! Точек: 4
3. Тест ошибок:
   Мало точек: Количество точек должно быть не менее 2
   Неупорядоченные точки: Точки должны быть строго упорядочены по возрастанию X


=== ЗАДАНИЕ 2: ИНТЕРФЕЙС FUNCTION И НАСЛЕДОВАНИЕ ===
1. Проверка наследования:
   ArrayTabulatedFunction instanceof Function: true
   LinkedListTabulatedFunction instanceof Function: true
2. Тест методов интерфейса Function:
   Array - границы: [0.0, 2.0]
   Array - f(1.5) = 2.5
   LinkedList - границы: [0.0, 2.0]
   LinkedList - f(1.5) = 2.5


=== ЗАДАНИЕ 3: АНАЛИТИЧЕСКИЕ ФУНКЦИИ ===
1. Экспонента:
   Границы: [-Infinity, Infinity]
   exp(0) = 1.0
   exp(1) = 2.718281828459045
   exp(-1) = 0.36787944117144233
```

```
2. Логарифм по основанию 2:
   Границы: [0.0, Infinity]
   log2(1) = 0.0
   log2(2) = 1.0
   log2(4) = 2.0
   log2(0) = NaN

3. Тригонометрические функции:
   sin(0) = 0.0
   cos(0) = 1.0
   tan(0) = 0.0
   sin(PI/2) = 1.0
   cos(PI/2) = 6.123233995736766E-17

=== ЗАДАНИЕ 4: КОМБИНИРОВАННЫЕ ФУНКЦИИ ===
1. Сумма sin + cos:
   Границы: [-Infinity, Infinity]
   (sin+cos)(0) = 1.0

2. Произведение sin * cos:
   (sin*cos)(0) = 0.0
   (sin*cos)(PI/4) = 0.5

3. Квадрат синуса:
   sin^2(0) = 0.0
   sin^2(PI/2) = 1.0

4. Масштабированный синус:
   scaledSin(PI) = 3.0
```

```
5. Сдвинутый синус:
   shiftedSin(0) = 0.0

6. Композиция exp(sin(x)):
   exp(sin(0)) = 1.0
   exp(sin(PI/2)) = 2.718281828459045

=== ЗАДАНИЕ 5: УТИЛИТНЫЙ КЛАСС FUNCTIONS ===
1. Тест shift:
   shifted(0) = 1.1585290151921035

2. Тест scale:
   scaled(PI/2) = 2.1213203435596424

3. Тест power:
   squared(PI/2) = 1.0

4. Тест sum:
   sum(0) = 1.0

5. Тест mult:
   product(PI/4) = 0.5

6. Тест composition:
   composed(0) = 1.0

7. Приватный конструктор:
   Создание объектов Functions запрещено - корректно
```

```
=== ЗАДАНИЕ 6: ТАБУЛИРОВАНИЕ ФУНКЦИЙ ===
1. Табуляция синуса на [0, PI] с 5 точками:
   Количество точек: 5
   Границы: [0.0, 3.141592653589793]
   Точки функции:
     [0] x=0,000, y=0,000
     [1] x=0,785, y=0,707
     [2] x=1,571, y=1,000
     [3] x=2,356, y=0,707
     [4] x=3,142, y=0,000

2. Сравнение с аналитической функцией:
   sin(PI/4): аналитическое=0,707107, табулированное=0,707107

3. Тест ошибок:
   Выход за границы: Левая граница табулирования -1.0 выходит за левую границу области определения 0.0
   Мало точек: Количество точек должно быть не менее 2
   Неправильные границы: Левая граница должна быть меньше правой

=== ЗАДАНИЕ 7: ВВОД-ВЫВОД ФУНКЦИЙ ===
Исходная функция (5 точек синуса на [0, PI]):
  [0] x=0,000, y=0,000
  [1] x=0,785, y=0,707
  [2] x=1,571, y=1,000
  [3] x=2,356, y=0,707
  [4] x=3,142, y=0,000

1. Тест байтового ввода/вывода:
   Данные записаны в байтовый поток, размер: 84 байт
```

```
   Функция прочитана из байтового потока
   Количество точек: 5
   Функции идентичны: true

2. Тест символьного ввода/вывода:
   Данные записаны в строку: "5 0.0 0.0 0.7853981633974483
0.7071067811865475 1.5707963267948966 1.0 2.356194490192345
0.7071067811865476 3.141592653589793 0.0"
   Функция прочитана из строки
   Количество точек: 5
   Функции идентичны: true

=== ЗАДАНИЕ 8: КОМПЛЕКСНОЕ ТЕСТИРОВАНИЕ ===
1. ТЕСТ АНАЛИТИЧЕСКИХ ФУНКЦИЙ:
   Значения sin и cos на [0, PI] с шагом 0.5:
   x             sin(x)          cos(x)
   0,0           0,000000        1,000000
   0,5           0,479426        0,877583
   1,0           0,841471        0,540302
   1,5           0,997495        0,070737
   2,0           0,909297        -0,416147
   2,5           0,598472        -0,801144
   3,0           0,141120        -0,989992

2. ТЕСТ ТАБУЛИРОВАННЫХ АНАЛОГОВ:
   Сравнение в точках табуляции:
   x             sin(x)          tab_sin(x)      cos(x)
tab_cos(x)
   0,0           0,000000        0,000000        1,000000
1,000000
```

```
   0,6          0,587785       0,587785       0,809017
0,809017
   1,3          0,951057       0,951057       0,309017
0,309017
   1,9          0,951057       0,951057       -0,309017      -
0,309017
   2,5          0,587785       0,587785       -0,809017      -
0,809017
   3,1          0,000000       0,000000       -1,000000      -
1,000000


3. ТЕСТ СУММЫ КВАДРАТОВ:
   sin(x)^2 + cos(x)^2 для разных количеств точек:
   Точки: 5
     x=0,0: 1,000000
     x=1,0: 0,883675
     x=2,0: 0,854819
     x=3,0: 0,913432
   Точки: 10
     x=0,0: 1,000000
     x=1,0: 0,985897
     x=2,0: 0,976203
     x=3,0: 0,970920
   Точки: 20
     x=0,0: 1,000000
     x=1,0: 0,998756
     x=2,0: 0,997638
     x=3,0: 0,996644

4. ТЕСТ ФАЙЛОВЫХ ОПЕРАЦИЙ:
   Экспонента записана в exp_function.txt
```

```
   Экспонента прочитана из файла
   Сравнение значений:
     x=0: исходная=1,000000, прочитанная=1,000000
     x=2: исходная=7,389056, прочитанная=7,389056
     x=4: исходная=54,598150, прочитанная=54,598150
     x=6: исходная=403,428793, прочитанная=403,428793
     x=8: исходная=2980,957987, прочитанная=2980,957987
     x=10: исходная=22026,465795, прочитанная=22026,465795
   Логарифм записан в log_function.bin
   Логарифм прочитан из файла

=== ЗАДАНИЕ 9: СЕРИАЛИЗАЦИЯ ===
Исходная функция ln(exp(x)) на [0, 10]:
  [0] x=0,0, y=NaN
  [1] x=1,0, y=1,000000
  [2] x=2,0, y=2,000000
  [3] x=3,0, y=3,000000
  [4] x=4,0, y=4,000000
  [5] x=5,0, y=5,000000
  [6] x=6,0, y=6,000000
  [7] x=7,0, y=7,000000
  [8] x=8,0, y=8,000000
  [9] x=9,0, y=9,000000
  [10] x=10,0, y=10,000000

1. СЕРИАЛИЗАЦИЯ С Serializable:
   Записано в function_serializable.ser
   Прочитано из function_serializable.ser
   Размер файла: 236 байт
   Функция корректна: true
```

2. СЕРИАЛИЗАЦИЯ С Externalizable:
   Записано в function_externalizable.ser
   Прочитано из function_externalizable.ser
   Размер файла: 236 байт
   Функция корректна: true

3. СРАВНЕНИЕ ФАЙЛОВ:
   Serializable: 236 байт
   Externalizable: 236 байт
   Текстовый: 235 байт
   Бинарный: 164 байт

4. СРАВНЕНИЕ ЗНАЧЕНИЙ ПОСЛЕ ДЕСЕРИАЛИЗАЦИИ:

x	Исходная		Serializable	Externalizable
0	NaN	NaN	NaN	
1	1,000000		1,000000	1,000000
2	2,000000		2,000000	2,000000
3	3,000000		3,000000	3,000000
4	4,000000		4,000000	4,000000
5	5,000000		5,000000	5,000000
6	6,000000		6,000000	6,000000
7	7,000000		7,000000	7,000000
8	8,000000		8,000000	8,000000
9	9,000000		9,000000	9,000000
10	10,000000		10,000000	10,000000