Лабораторная работа №3

Разработка классов табулированных функций и обработка исключений

Выполнил: студент

Группы 6301-030301D

Дымченко Владислав Романович

Задание 1: Ознакомление с классами исключений

Изучили документацию к стандартным классам исключений Java: Exception, IndexOutOfBoundsException, ArrayIndexOutOfBoundsException, IllegalArgumentException, IllegalStateException.

Задание 2: Создание классов исключений

Созданы два класса исключений в пакете functions:

FunctionPointIndexOutOfBoundsException – исключение выхода за границы набора точек при обращении к ним по номеру, наследует от класса IndexOutOfBoundsException;

InappropriateFunctionPointException – исключение, выбрасываемое при попытке добавления или изменения точки функции несоответствующим образом, наследует от класса Exception.

Задание 3: Изменение TabulatedFunction

Класс TabulatedFunction изменен и переименован в ArrayTabulatedFunction:

- Оба конструктора класса должны выбрасывают исключение IllegalArgumentException, если левая граница области определения больше или равна правой, а также если предлагаемое количество точек меньше двух.

- Методы работы с точками выбрасывают FunctionPointIndexOutOfBoundsException при неверных индексах

- Методы setPoint(), setPointX() и addPoint() выбрасывают InappropriateFunctionPointException при нарушении порядка точек

- Метод deletePoint() выбрасывает IllegalStateException при попытке удалить точку, если точек меньше трех.

Задание 4-5: LinkedListTabulatedFunction

Реализован класс LinkedListTabulatedFunction с использованием двусвязного циклического списка:

- Внутренний класс FunctionNode содержащий информационное поле для хранения данных типа FunctionPoint, а также поля для хранения ссылок на предыдущий и следующий элемент.
- Методы getNodeByIndex(), addNodeToTail(), addNodeByIndex(), deleteNodeByIndex() для работы со списком
- Оптимизированный доступ к элементам через кэширование последнего accessed узла
- Реализованы в классе конструкторы и методы, аналогичные конструкторам и методам класса TabulatedFunction. Конструкторы имеют те же параметры, методы имеют те же сигнатуры. Также выбрасываются те же виды исключений в тех же случаях.

Задание 6: Создание интерфейса

Создан интерфейс TabulatedFunction, содержащий объявления общих методов
классов ArrayTabulatedFunction и LinkedListTabulatedFunction.
-Методы работы с точками (getPoint, setPoint, addPoint, deletePoint)
- Методы получения координат (getPointX, getPointY, setPointX, setPointY)
- Методы получения информации о функции (getPointsCount, getLeftDomainBorder, getRightDomainBorder, getFunctionValue)

Задание 7: Проверка работы написанных классов.

Разработан класс Main для тестирования всех функций и обработки исключений. Для этого в созданном ранее классе Main, содержащем точку входа программы, добавили проверку для случаев, в которых объект табулированной функции должен выбрасывать исключения.

Внесенные изменения и доработки

1. Оптимизация метода getFunctionValue()

Добавлена проверка точного совпадения X координаты перед интерполяцией.

В ArrayTabulatedFunction.java раньше сразу искал интервал для интерполяции, теперь, сначала проверял точное совпадение с существующими точками, если его нету – ищу интервал для интерполяции.

```
public double getFunctionValue(double x) {  8 usages   Vlados-ux
    if (x < getLeftDomainBorder() - EPSILON || x > getRightDomainBorder() + EPSILON) {
        return Double.NaN;
    }

    for (int i = 0; i < pointsCount; i++) {
        if (Math.abs(points[i].getX() - x) < EPSILON) {
            return points[i].getY();
        }
    }
```

В LinkedListTabulatedFunction.java аналогично:

```
public double getFunctionValue(double x) {  8 usages   Vlados-ux
    if (x < getLeftDomainBorder() -EPSILON || x > getRightDomainBorder() + EPSILON) {
        return Double.NaN;
    }

    FunctionNode node = head.next;
    while (node != head) {
        if (Math.abs(node.point.getX() - x) < EPSILON) {
            return node.point.getY();
        }
        node = node.next;
    }
```

В результате этих изменений мы устранили избыточные вычисления (не выполняем интерполяцию при точном совпадении), повысили точность (возвращаем точное значение Y вместо интерполированного).

2.Исправление тестирования Main для Array

Была произведена замена бессмысленного изменения ординаты на осмысленный тест в методе testArrayTabulatedFunction()

```
System.out.println("Исправление некорректного значения ординаты:");
System.out.printf("До изменения: (%.1f; %.1f) ", func.getPointX( index: 2), func.getPointY( index: 2));

func.setPointY( index: 2, y: 100.0);
System.out.printf("После установки 100: (%.1f; %.1f) ", func.getPointX( index: 2), func.getPointY( index: 2));

double correctY = func.getPointX( index: 2) + 5.0;
func.setPointY( index: 2, correctY);
System.out.printf("После исправления: (%.1f; %.1f)\n", func.getPointX( index: 2), func.getPointY( index: 2));
```

Это изменение дало осмысленный тест - показываем испорченные данные и их исправление.

3.**Добавление демонстрации точек для LinkedList.**

Добавлен вывод всех точек после каждой операции для LinkedList

В Main.java в методе testLinkedListTabulatedFunction()

```
System.out.println("\nТочки перед операциями:");
for(int i = 0; i < func.getPointsCount(); ++i) {
    FunctionPoint p = func.getPoint(i);
    System.out.printf("(%.1f; %.1f) ", p.getX(), p.getY());
}
System.out.println();
```

```
System.out.println("Точки после добавления (3.3, 8.3):");
for(int i = 0; i < func.getPointsCount(); ++i) {
    FunctionPoint p = func.getPoint(i);
    System.out.printf("(%.1f; %.1f) ", p.getX(), p.getY());
}
System.out.println();
```

```
System.out.println("Точки после удаления точки с индексом 5:");
for(int i = 0; i < func.getPointsCount(); ++i) {
    FunctionPoint p = func.getPoint(i);
    System.out.printf("(%.1f; %.1f) ", p.getX(), p.getY());
}
System.out.println();
```

4.Дополнительное тестирование точного совпадения

Новый тест для проверки оптимизации getFunctionValue() в Main.java в методе testLinkedListTabulatedFunction().

```
System.out.println("\nПроверка точного совпадения:");
double exactX = 3.0;
double exactY = func.getFunctionValue(exactX);
System.out.printf("f(%.1f) = %.1f (должно быть точно 8.0 без интерполяции)\n", exactX, exactY);
```

Результаты тестирования

Тестирование классов табулированной функции

Тестирование ArrayTabulatedFunction (f(x) = x + 5):
Область определения: [0,0, 10,0]
Кол-во точек: 11

Точки табулированной функции:
Точка 0: (0,0; 5,0)
Точка 1: (1,0; 6,0)
Точка 2: (2,0; 7,0)
Точка 3: (3,0; 8,0)
Точка 4: (4,0; 9,0)
Точка 5: (5,0; 10,0)
Точка 6: (6,0; 11,0)
Точка 7: (7,0; 12,0)
Точка 8: (8,0; 13,0)
Точка 9: (9,0; 14,0)
Точка 10: (10,0; 15,0)

Вычисленные значения функции:
f(-5,0) = не определено (не входит в область определения)
f(-2,0) = не определено (не входит в область определения)
f(0,0) = 5,0 (должно быть: 5,0)
f(1,0) = 6,0 (должно быть: 6,0)
f(2,5) = 7,5 (должно быть: 7,5)
f(3,0) = 8,0 (должно быть: 8,0)
f(4,5) = 9,5 (должно быть: 9,5)

f(5,0) = 10,0 (должно быть: 10,0)
f(6,5) = 11,5 (должно быть: 11,5)
f(7,0) = 12,0 (должно быть: 12,0)
f(8,5) = 13,5 (должно быть: 13,5)
f(10,0) = 15,0 (должно быть: 15,0)
f(12,0) = не определено (не входит в область определения)
f(15,0) = не определено (не входит в область определения)

Проверка модификации точек:
Исправление некорректного значения ординаты:
До изменения: (2,0; 7,0) После установки 100: (2,0; 100,0) После исправления: (2,0; 7,0)

Корректное изменение абсциссы с обновлением Y:
До: (4,0; 9,0) После: (4,2; 9,2)

Добавление точек:
Количество точек до добавления: 11
Количество точек после добавления: 12
Значение в добавленной точке: f(3,3) = 8,3
Точки после добавления:
(0,0; 5,0) (1,0; 6,0) (2,0; 7,0) (3,0; 8,0) (3,3; 8,3) (4,2; 9,2) (5,0; 10,0) (6,0; 11,0) (7,0; 12,0) (8,0; 13,0) (9,0; 14,0) (10,0; 15,0)

Удаление точек:
Количество точек до удаления: 12
Количество точек после удаления: 11

Обновление всех значений y в соответсвии с f(x) = x + 5:
Точка 0: (0,0; 5,0)
Точка 1: (1,0; 6,0)
Точка 2: (3,0; 8,0)
Точка 3: (3,3; 8,3)
Точка 4: (4,2; 9,2)
Точка 5: (5,0; 10,0)
Точка 6: (6,0; 11,0)

Точка 7: (7,0; 12,0)
Точка 8: (8,0; 13,0)
Точка 9: (9,0; 14,0)
Точка 10: (10,0; 15,0)

Интерполяция:
f(0,3) = 5,3 (должно быть: 5,3) Правильно
f(1,1) = 6,1 (должно быть: 6,1) Правильно
f(2,8) = 7,8 (должно быть: 7,8) Правильно
f(3,9) = 8,9 (должно быть: 8,9) Правильно
f(5,2) = 10,2 (должно быть: 10,2) Правильно
f(6,7) = 11,7 (должно быть: 11,7) Правильно
f(8,1) = 13,1 (должно быть: 13,1) Правильно
f(9,4) = 14,4 (должно быть: 14,4) Правильно

Границы:
Левая граница: f(0,0) = 5,0
Правая граница: f(10,0) = 15,0

==========
==========

Тестирование LinkedListTabulatedFunction (f(x) = x + 5):
Область определения: [0,0, 10,0]
Кол-во точек: 11

Точки перед операциями:
(0,0; 5,0) (1,0; 6,0) (2,0; 7,0) (3,0; 8,0) (4,0; 9,0) (5,0; 10,0) (6,0; 11,0)
(7,0; 12,0) (8,0; 13,0) (9,0; 14,0) (10,0; 15,0)

Вычисленные значения функции:
f(-5,0) = не определено (не входит в область определения)
f(-2,0) = не определено (не входит в область определения)
f(0,0) = 5,0 (должно быть: 5,0)
f(1,0) = 6,0 (должно быть: 6,0)
f(2,5) = 7,5 (должно быть: 7,5)

f(3,0) = 8,0 (должно быть: 8,0)
f(4,5) = 9,5 (должно быть: 9,5)
f(5,0) = 10,0 (должно быть: 10,0)
f(6,5) = 11,5 (должно быть: 11,5)
f(7,0) = 12,0 (должно быть: 12,0)
f(8,5) = 13,5 (должно быть: 13,5)
f(10,0) = 15,0 (должно быть: 15,0)
f(12,0) = не определено (не входит в область определения)
f(15,0) = не определено (не входит в область определения)

Тестирование добавления и удаления:
Количество точек до добавления: 11
Количество точек после добавления: 12
Точки после добавления (3.3, 8.3):
(0,0; 5,0) (1,0; 6,0) (2,0; 7,0) (3,0; 8,0) (3,3; 8,3) (4,0; 9,0) (5,0; 10,0) (6,0; 11,0) (7,0; 12,0) (8,0; 13,0) (9,0; 14,0) (10,0; 15,0)
Количество точек после удаления: 11
Точки после удаления точки с индексом 5:
(0,0; 5,0) (1,0; 6,0) (2,0; 7,0) (3,0; 8,0) (3,3; 8,3) (5,0; 10,0) (6,0; 11,0) (7,0; 12,0) (8,0; 13,0) (9,0; 14,0) (10,0; 15,0)

Проверка интерполяции:
f(0,3) = 5,3 (ожидалось: 5,3) ✓
f(1,1) = 6,1 (ожидалось: 6,1) ✓
f(2,8) = 7,8 (ожидалось: 7,8) ✓
f(3,9) = 8,9 (ожидалось: 8,9) ✓
f(5,2) = 10,2 (ожидалось: 10,2) ✓
f(6,7) = 11,7 (ожидалось: 11,7) ✓
f(8,1) = 13,1 (ожидалось: 13,1) ✓
f(9,4) = 14,4 (ожидалось: 14,4) ✓

Проверка точного совпадения:
f(3,0) = 8,0 (должно быть точно 8.0 без интерполяции)

==========
==========

Тестирование исключений:

Неверные параметры конструктора:
Исключение выброшено: Левая граница должна быть меньше правой
Исключение выброшено: Количество точек должно быть не менее 2

Выход за границы индекса:
Исключение выброшено: Индекс 10 вне границ [0, 2]

Нарушение порядка точек:
Исключение выброшено: Новая координата X нарушает порядок точек

Дублирование x координат:
Исключение выброшено: Точка с X=2.5 уже существует

Удаление при минимальном количестве точек:
Исключение выброшено: Нельзя удалить точку: минимальное количество точек - 2