ЛАБОРАТОРНАЯ РАБОТА №3

Табулированные функции: реализация на массивах и связных списках

по курсу
Объектно-ориентированное программирование

Группа 6204-010302D

Студент: Е.Д.Васильев.

Преподаватель: Борисов Дмитрий Сергеевич.

Задание 1

Изучение классов исключений входящих в API Java:

Были изучены следующие классы исключений Java:

- **Exception** - базовый класс всех проверяемых исключений
- **IndexOutOfBoundsException** - недопустимый индекс(например, вне границ массива или списка)
- **ArrayIndexOutOfBoundsException** - частный случай предыдущего класса, выход за границы массива
- **IllegalArgumentException** - неверный аргумент метода
- **IllegalStateException** - непроверяемое исключение, возникает при недопустимом состоянии объекта

Задание 2

Создание пользовательских исключений

В пакет functions были добавлены два новых класса исключений:

1) FunctionPointIndexOutOfBoundsException

Наследуется от IndexOutOfBoundsException. Выбрасывается, если происходит обращение к точке функции по индексу, выходящему за границы массива или списка точек.

FunctionPointIndexOutOfBoundsException.java:

```
package functions;

public class FunctionPointIndexOutOfBoundsException extends
IndexOutOfBoundsException{

        // Конструктор без параметров
        public FunctionPointIndexOutOfBoundsException() {
            super();
        }

        // Конструктор с сообщением
        public FunctionPointIndexOutOfBoundsException(String message) {
            super(message);
        }
    }
```

2) InappropriateFunctionPointException

Наследуется от Exception (checked exception). Используется, если новая точка нарушает упорядоченность X или если добавляется точка с уже существующими координатами X.

InappropriateFunctionPointException.java:

```
package functions;

public class InappropriateFunctionPointException extends Exception{
    // Конструктор без параметров
    public InappropriateFunctionPointException() {
        super();
    }

    // Конструктор с сообщением
    public InappropriateFunctionPointException(String message) {
        super(message);
    }
}
```

Задание 3
Модификация TabulatedFunction

В класс TabulatedFunction были внесены изменения, связанные с исключениями.

Изменения в конструкторах:

Конструкторы теперь выбрасывают IllegalArgumentException, если:

- левая граница leftX больше или равна правой rightX;
- количество точек < 2.

1. **Конструктор с равномерными интервалами по X:**

```
public ArrayTabulatedFunction(double leftX, double rightX, int pointsCount) {

    if (leftX >= rightX - EPS) {
        throw new IllegalArgumentException("Left border must be less than
right border");
    }

    if (pointsCount < 2) {
        throw new IllegalArgumentException("At least two points required");
    }
```

2. **Конструктор с заданными значениями:**

```
public ArrayTabulatedFunction(double leftX, double rightX, double[] values) {

    if (leftX >= rightX - EPS) {
        throw new IllegalArgumentException("Left border must be less than
right border");
```

```
    }

    if (pointsCount < 2) {
        throw new IllegalArgumentException("At least two points required");
```

Изменённые методы (добавлено выбрасывание исключений)

В классе **ArrayTabulatedFunction** были изменены методы, чтобы они корректно выбрасывали собственные исключения.

1. **Методы getPoint, getPointX, getPointY**
   Теперь выбрасывают исключение **FunctionPointIndexOutOfBoundsException**, если обращение к точке происходит по индексу, который выходит за границы допустимого диапазона.
2. **Методы setPoint и setPointX**
   Теперь выбрасывают два типа исключений:
   - **FunctionPointIndexOutOfBoundsException** — если индекс точки некорректен;
   - **InappropriateFunctionPointException** — если новое значение X нарушает порядок точек по возрастанию.
3. **Метод deletePoint**
   Теперь выбрасывает два вида исключений:
   - **FunctionPointIndexOutOfBoundsException** — если удаление выполняется по неверному индексу;
   - **IllegalStateException** — если попытаться удалить точку так, что в функции останется меньше двух точек.
4. **Метод addPoint**
   Теперь выбрасывает исключение **InappropriateFunctionPointException**, если добавляемая точка имеет X, совпадающий с уже существующей точкой — что нарушает строгое упорядочивание.

Задание 4-5

Реализация связного списка

Создан новый класс LinkedListTabulatedFunction, использующий двусвязный циклический список с выделенной головой.

Новый внутренний класс FunctionNode:

```
public class LinkedListTabulatedFunction{

    private static class FunctionNode {
        FunctionPoint point;
        FunctionNode next;
        FunctionNode prev;

        FunctionNode(FunctionPoint point) {
            this.point = point;
        }
```

Основные методы, реализующие работу с узлами:

Основные методы, реализующие работу с узлами:

- **getNodeByIndex(int index)** - возвращает узел по индексу с оптимизацией поиска (с начала или с конца)
- **addNodeToTail()** - добавляет узел в конец списка
- **addNodeByIndex(int index) -** вставляет узел в произвольное место\
- **deleteNodeByIndex(int index) –** удаляет узел по индексу

Реализация методов табулированной функции:

Методы аналогичны ArrayTabulatedFunction, но работают со списком: getPoint, setPoint, setPointX, setPointY, addPoint, deletePoint, getFunctionValue.

Важное отличие от **TabulatedFunction** заключается в том, что удаление и вставка выполняются без сдвига элементов, только изменением ссылок prev и next.

Задание 6

Создание интерфейса

Класс TabulatedFunction переименован в ArrayTabulatedFunction. Создан интерфейс TabulatedFunction, который теперь реализуют оба класса:

```java
package functions;

public interface TabulatedFunction {

    // Возвращает количество точек
    int getPointsCount();

    // Возвращает левую границу
    double getLeftDomainBorder();

    // Возвращает правую границу
    double getRightDomainBorder();

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

    // Возвращает точку по индексу
    FunctionPoint getPoint(int index) throws
FunctionPointIndexOutOfBoundsException;

    // Заменяет точку по индексу
    void setPoint(int index, FunctionPoint point) throws
FunctionPointIndexOutOfBoundsException, InappropriateFunctionPointException;

    // Возвращает X точки по индексу
    double getPointX(int index) throws
FunctionPointIndexOutOfBoundsException;

    // Устанавливает X точки по индексу
    void setPointX(int index, double x) throws
FunctionPointIndexOutOfBoundsException, InappropriateFunctionPointException;

    // Возвращает Y точки по индексу
    double getPointY(int index) throws
FunctionPointIndexOutOfBoundsException;

    // Устанавливает Y точки по индексу
    void setPointY(int index, double y) throws
FunctionPointIndexOutOfBoundsException;

    // Удаляет точку по индексу
    void deletePoint(int index) throws
FunctionPointIndexOutOfBoundsException, IllegalStateException;

    // Добавляет новую точку (сохраняя порядок)
    void addPoint(FunctionPoint point) throws
InappropriateFunctionPointException;

}
```

Задание 7

Тестирование

Создан класс Main для тестирования:
Исходные точки: x = -1.0, y = 0.0 x = -0.5, y = 0.0 x = 0.0, y = 0.0 x = 0.5, y = 0.0 x = 1.0, y = 0.0 После замены точки:

x = -1.0, y = 0.0
x = -0.5, y = 0.0
x = 0.0, y = 0.0 x
= 0.55, y = 0.25 x
= 1.0, y = 0.0
После
добавления
точки: x = -1.0, y
= 0.0 x = -0.5, y
= 0.0 x = 0.0, y =
0.0 x = 0.55, y =
0.25 x = 0.7, y =
0.3 x = 1.0, y =
0.0 После
удаления точки:
x = -1.0, y = 0.0
x = -0.5, y = 0.0
x = 0.0, y = 0.0
x = 0.7, y = 0.3
x = 1.0, y = 0.0
Поймано исключение: functions.FunctionPointIndexOutOfBoundsException
Поймано исключение: functions.InappropriateFunctionPointException
Поймано исключение: functions.InappropriateFunctionPointException
Исходные точки: x = -1.0, y = 0.0 x = -0.5, y = 0.0 x = 0.0, y = 0.0 x =
0.5, y = 0.0 x = 1.0, y = 0.0 После замены точки:
x = -1.0, y = 0.0 x = -0.5,
y = 0.0 x = 0.0, y = 0.0 x
= 0.55, y = 0.25 x = 1.0, y
= 0.0 После добавления
точки: x = -1.0, y = 0.0 x
= -0.5, y = 0.0 x = 0.0, y =
0.0 x = 0.55, y = 0.25 x =
0.7, y = 0.3 x = 1.0, y =
0.0 После удаления
точки:
x = -1.0, y = 0.0
x = -0.5, y = 0.0
x = 0.0, y = 0.0
x = 0.7, y = 0.3
x = 1.0, y = 0.0

f(-0.8) = 0.0 f(-
0.2) = 0.0 f(0.2)
=
0.08571428571
428572 f(0.6) =
0.25714285714
28572
f(1.5) Поймано исключение: functions.InappropriateFunctionPointException
Поймано исключение: functions.FunctionPointIndexOutOfBoundsException
Поймано исключение: functions.InappropriateFunctionPointException
Поймано исключение: functions.InappropriateFunctionPointException