Лабораторная №3

Студент: Гонеев Роман Олегович
Группа: 6201-120303D

Задания:

1. -
2. В пакете functions создал два класса исключений:

   cd ~/lab3_rep/functions

- FunctionPointIndexOutOfBoundsException:

  nano FunctionPointIndexOutOfBoundsException.java

  Наследует от класса IndexOutOfBoundsException:

  ```
  class FunctionPointIndexOutOfBoundsException extends IndexOutOfBoundsException
  ```

- InappropriateFunctionPointException:

  nano InappropriateFunctionPointException.java

  Наследует от класса Exception:

  ```
  public class InappropriateFunctionPointException extends RuntimeException
  ```

3. В классе TabulatedFunction добавил исключения методами класса.
   - IllegalArgumentException для проверки границ
   - FunctionPointIndexOutOfBoundsException индекс вне границ набора точек
   - InappropriateFunctionPointreportxception Абсцисса нарушает упорядочность или абсцисса добавляемой совпадает с существующей
   - IllegalStateException Количество точек меньше трех
4. В пакете functions создал класс LinkedListTabulatedFunction, использующий двувсязный список с выделнной головой для хранения точек.
   Для реализации списка определил два класса:
   - FunctionNode — описывает элемент списка. Для обеспечения инкапсуляции класс описан как вложеный приватный.
     В классе LinkedListTabulatedFunction реализовал вспомогательные методы для работы со структурой списка:
     1. FunctionNode getNodeByIndex(int index) — возвращает ссылку ну зел по его номеру
     2. addPoint(FunctionPoint point) — вставляет точку, соблюдая упорядоченность.
   - LinkedListTabulatedFunction — внешний класс. Описывает список целиком, храня ссылку на голову и размер.
     Конструкторы и методы класса LinkedListTabulatedFunction реализованы аналогично ArrayTabulatedFunction, с теми же видами исключений. Логика методов getPoint, setPointX, deletePoint использует реализованные методы работы со связным списком.
5. Реализация функционала TabulatedFunction в LinkedListTabulatedFunction
   В классе реализовал два конструктора, аналогичные конструкторам ArrayTabulatedFunction, и обеспечивающие те же проверки на корректность входных данных:

   - LinkedListTabulatedFunction(double leftX, double rightX, int pointsCount)

- LinkedListTabulatedFunction(double leftX, double rightX, double[] values)

оба выбрасывают исключения IllegalArgumentException в случае некорректных параметров. Инициализация через addNodeTail.

Реализация методов и оптимизация:

- getPointX(int index) и getPointY(int index) — использует getNodeByIndex(index) для доступа к узлу, потом напрямую возвращает x или y из поля data
- setPointX(int index, double x) — получает узел через getNodeByIndex(index); проверяет новое значение x, обращаясь напрямую к полям соседних узлов (prev.data.getX() и next.data.getX()) - это является оптимизацией для связного списка.
- deletePoint(int index) - использует deleteNodeByIndex(index) для удаления узла путем пересвязывания ссылок prev и next у соседей.
- addPoint(FunctionPoint point) - осуществляет проход по списку для проверки на дубликат x и нахождения места для вставки. Вставка точки производится путем переназначения ссылок у соседних узлов.

6. Провел реструктуризацию классов для внедрения концепции полиморфизма.
   Класс TabulatedFunction был переименован в ArrayTabulatedFunction. Был создан интерфейс TabulatedFunction, который определяет шаблон(«контракт») для всех реализаций (массив и список):
   public interface TabulatedFunction {}
   Оба класса, ArrayTabulatedFunction и LinkedListTabulatedFunction, были модифицированы для реализации этого интерфейса с помощью ключевого слова
   - public class ArrayTabulatedFunction implements TabulatedFunction {}
   - public class LinkedListTabulatedFunction implements TabulatedFunction {}
7. Проверка работы написанных классов

Проверил корректность реализации классов ArrayTabulatedFunction и LinkedListTabulatedFunction и их соответствие интерфейсу TabulatedFunction.

1. [1] Тест полиморфизма:
   Интерполяция (x=2,5): Array=21,25, List=21,25
   Сравнение результатов Array и List: Ок
   Проверка значения (Ожидается 21.25): Ок
   [2] Тест операций: добавление, удаление, изменение

2. 2.1. setPointY (Индекс 2: X=30) --
   Состояние до setPointY (Size: 4):
   [0]: X=10,00, Y=100,00
   [1]: X=20,00, Y=200,00
   [2]: X=30,00, Y=300,00
   [3]: X=40,00, Y=400,00
   Состояние после setPointY (Y=333.33) (Size: 4):
   [0]: X=10,00, Y=100,00
   [1]: X=20,00, Y=200,00
   [2]: X=30,00, Y=333,33
   [3]: X=40,00, Y=400,00

2.2. addPoint (X=25.0, Y=250.0)
Состояние до addPoint (Size: 4):
[0]: X=10,00, Y=100,00
[1]: X=20,00, Y=200,00
[2]: X=30,00, Y=333,33
[3]: X=40,00, Y=400,00
Состояние после addPoint (X=25.0 вставлен) (Size: 5):
[0]: X=10,00, Y=100,00
[1]: X=20,00, Y=200,00
[2]: X=25,00, Y=250,00
[3]: X=30,00, Y=333,33
[4]: X=40,00, Y=400,00

2.3. deletePoint (Индекс 0: X=10.0) --
Состояние до deletePoint (Size: 5):
[0]: X=10,00, Y=100,00
[1]: X=20,00, Y=200,00
[2]: X=25,00, Y=250,00
[3]: X=30,00, Y=333,33
[4]: X=40,00, Y=400,00
Состояние после deletePoint (X=10.0 удален) (Size: 4):
[0]: X=20,00, Y=200,00
[1]: X=25,00, Y=250,00
[2]: X=30,00, Y=333,33
[3]: X=40,00, Y=400,00

3. [3] Тест исключений:
   -- 3.1. IllegalArgumentException --
   Ок: Пойман IllegalArgumentException (Границы).

   -- 3.2. FunctionPointIndexOutOfBoundsException --
   Ок: Пойман FunctionPointIndexOutOfBoundsException (Индекс).

   -- 3.3. InappropriateFunctionPointException (Дубликат X) --
   Ок: Пойман InappropriateFunctionPointException (Дубликат X).

   -- 3.4. InappropriateFunctionPointException (Нарушение порядка) --
   Ок: Пойман InappropriateFunctionPointException (Нарушение порядка).

   -- 3.5. IllegalStateException (Удаление при size < 3) --
   Ок: Пойман IllegalStateException (Size < 3).