Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное
образовательное учреждение высшего образования
«Самарский национальный исследовательский университет
имени академика С.П. Королева»

Институт информатики и кибернетики

Кафедра технической кибернетики

Отчет по лабораторной работе №2

Дисциплина: «ООП»

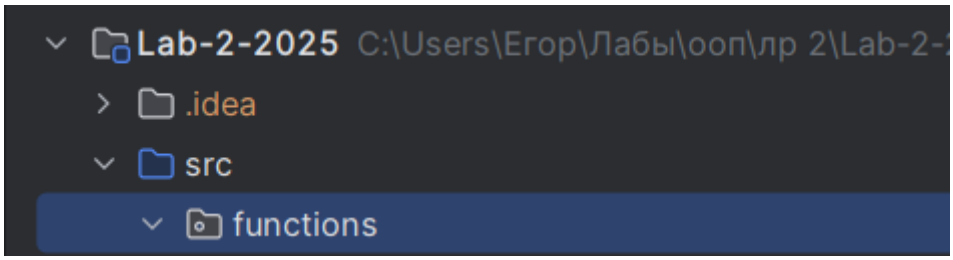
Выполнил: Илларионов Е.А.

Группа: 6201-120303D

Самара, 2025

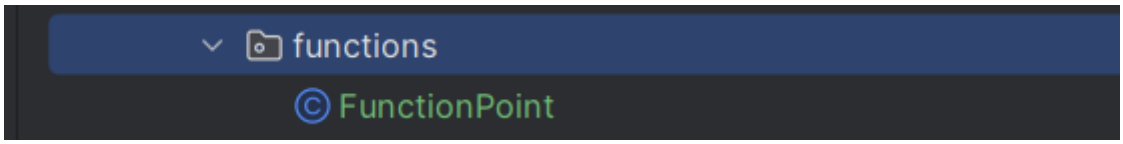
Задание 1

Создание пакета functions



Задание 2

В пакете functions создаём класс FunctionPoint. Пишем класс FunctionPoint, который будет описывать одну точку табулированной функции и хранить координаты x и y, предоставляя методы работы с ними.

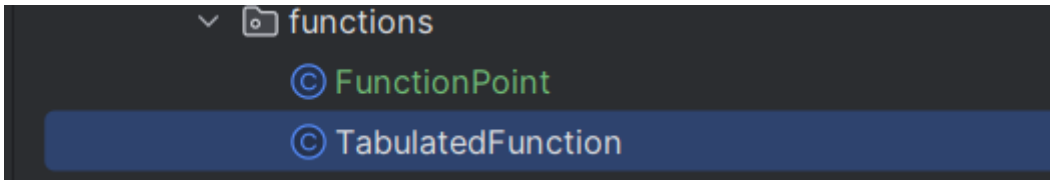


FunctionPoint.java × Main.java .gitignore TabulatedFunctio

```java
package functions;

public class FunctionPoint {
    private double x;
    private double y;

    public FunctionPoint(double x, double y) {
        this.x = x;
        this.y = y;
    }

    public FunctionPoint(FunctionPoint point) {
        this.x = point.x;
        this.y = point.y;
    }

    public FunctionPoint() {
        this.x = 0;
        this.y = 0;
    }

    public double getX() {
        return x;
    }

    public double getY() {
        return y;
    }

    public void setX(double x) {
        this.x = x;
    }

    public void setY(double y) {
        this.y = y;
    }
}
```

Задание 3

В пакете functions создаём класс TabulatedFunction



Пишем класс TabulatedFunction, который хранит и создаёт табулированную функцию в виде массива точек FunctionPoint, равномерно распределенных по оси x с заданными значениями y

TabulatedFunction.java | FunctionPoint.java | Main.java | .gitignore

```java
package functions;

public class TabulatedFunction {

    private FunctionPoint[] points;
    private int size;

    public TabulatedFunction(double leftX, double rightX, int pointsCount) {
        if (pointsCount < 2) throw new IllegalArgumentException("pointsCount must be >= 2");

        this.size = pointsCount;
        this.points = new FunctionPoint[pointsCount];
        double step = (rightX - leftX) / (pointsCount - 1);

        for (int i = 0; i < pointsCount; i++) {
            double x = leftX + i * step;
            this.points[i] = new FunctionPoint(x, y: 0);
        }
    }

    public TabulatedFunction(double leftX, double rightX, double[] values) {
        if (values.length < 2) throw new IllegalArgumentException("values length must be >= 2");

        this.size = values.length;
        this.points = new FunctionPoint[size];
        double step = (rightX - leftX) / (size - 1);

        for (int i = 0; i < size; i++) {
            double x = leftX + i * step;
            this.points[i] = new FunctionPoint(x, values[i]);
        }
    }
```

Задание 4

Реализованы методы getLeftDomainBorder() – возвращает минимальный x и getRightDomainBorder – возвращает максимальный x

getFunctionValue(double x) вычисляет значение с линейной интерполяцией

Задание 5

Добавлены методы для работы с точками, с учетом инкапсуляции и порядка

getPointsCount() Возвращает количество точек

getPoint(int index) Возвращает копию точки

setPoint(int index, FunctionPoint point) Заменяет точку на копию

setPointX(int index, double x) Изменяет абсциссу

setPointY(int index, double y) Изменяет ординату

```
60      public int getPointsCount() {
61          return size;
62      }
63
64      public FunctionPoint getPoint(int index) {
65          return new FunctionPoint(points[index]); // копия точки
66      }
67
68      public void setPoint(int index, FunctionPoint point) {
69          // Восстановлено: Проверка невалидного индекса (ИЛИ)
70          if (index < 0 || index >= size) return;
71
72          double newX = point.getX();
73          if (index > 0 && newX <= points[index - 1].getX()) return;
74          if (index < size - 1 && newX >= points[index + 1].getX()) return;
75
76          points[index] = new FunctionPoint(point);
77      }
78
79      public double getPointX(int index) {
80          return points[index].getX();
81      }
82
83      public void setPointX(int index, double x) {
84          // Восстановлено: Проверка невалидного индекса (ИЛИ)
85          if (index < 0 || index >= size) return;
86          if (index > 0 && x <= points[index - 1].getX()) return;
87          if (index < size - 1 && x >= points[index + 1].getX()) return;
88          points[index].setX(x);
89      }
```

```
90
91      public double getPointY(int index) {
92          return points[index].getY();
93      }
94
95      public void setPointY(int index, double y) {
96          points[index].setY(y);
97      }
```

Задание 6

Реализованы методы для изменения количества точек

deletePoint(int index) Удаляет точку сдвигом элементов

addPoint(FunctionPoint point) Добавляет точку в нужное место, сохраняя порядок по x

```
99      public void deletePoint(int index) {
100         // Восстановлено: Проверка размера массива ИЛИ невалидного индекса (ИЛИ)
101         if (size <= 2 || index < 0 || index >= size) return;
102
103         for (int i = index; i < size - 1; i++) {
104             points[i] = points[i + 1];
105         }
106         points[size - 1] = null;
107         size--;
108     }
109
110     public void addPoint(FunctionPoint point) {
111         int pos = 0;
112         while (pos < size && points[pos].getX() < point.getX()) pos++;
113
114         if (size == points.length) {
115             FunctionPoint[] newPoints = new FunctionPoint[size + 1];
116             System.arraycopy(points, srcPos: 0, newPoints, destPos: 0, pos);
117             newPoints[pos] = new FunctionPoint(point);
118             System.arraycopy(points, pos, newPoints, destPos: pos + 1, length: size - pos);
119             points = newPoints;
120         } else {
121             for (int i = size; i > pos; i--) {
122                 points[i] = points[i - 1];
123             }
124             points[pos] = new FunctionPoint(point);
125         }
126         size++;
127     }
128 }
```

Задание 7

Проверка работы написанных классов.

Создан класс Main для тестирования всех реализованных методов на примере функции $f(x) = x^2$

```
"C:\Program Files\Eclipse Adoptium\jdk-25.0.1.8-hotspot\bin\ja
Табулированная функция f(x) = x^2:
Точка 0: x=0.0, y=0.0
Точка 1: x=1.0, y=1.0
Точка 2: x=2.0, y=4.0
Точка 3: x=3.0, y=9.0
Точка 4: x=4.0, y=16.0

Значения функции в произвольных точках:
f(-1.0) = не определено
f(0.0) = 0.0
f(0.5) = 0.5
f(1.0) = 1.0
f(1.5) = 2.5
f(2.0) = 4.0
f(3.5) = 12.5
f(4.0) = 16.0
f(5.0) = не определено

Меняем y второй точки на 100:
f(1.0) = 100.0

Добавляем точку (2.5, 50):
Количество точек после добавления: 6

Удаляем точку с индексом 0:
Количество точек после удаления: 5
```

```
Все точки после изменений:
Точка 0: x=1.0, y=100.0
Точка 1: x=2.0, y=4.0
Точка 2: x=2.5, y=50.0
Точка 3: x=3.0, y=9.0
Точка 4: x=4.0, y=16.0
```