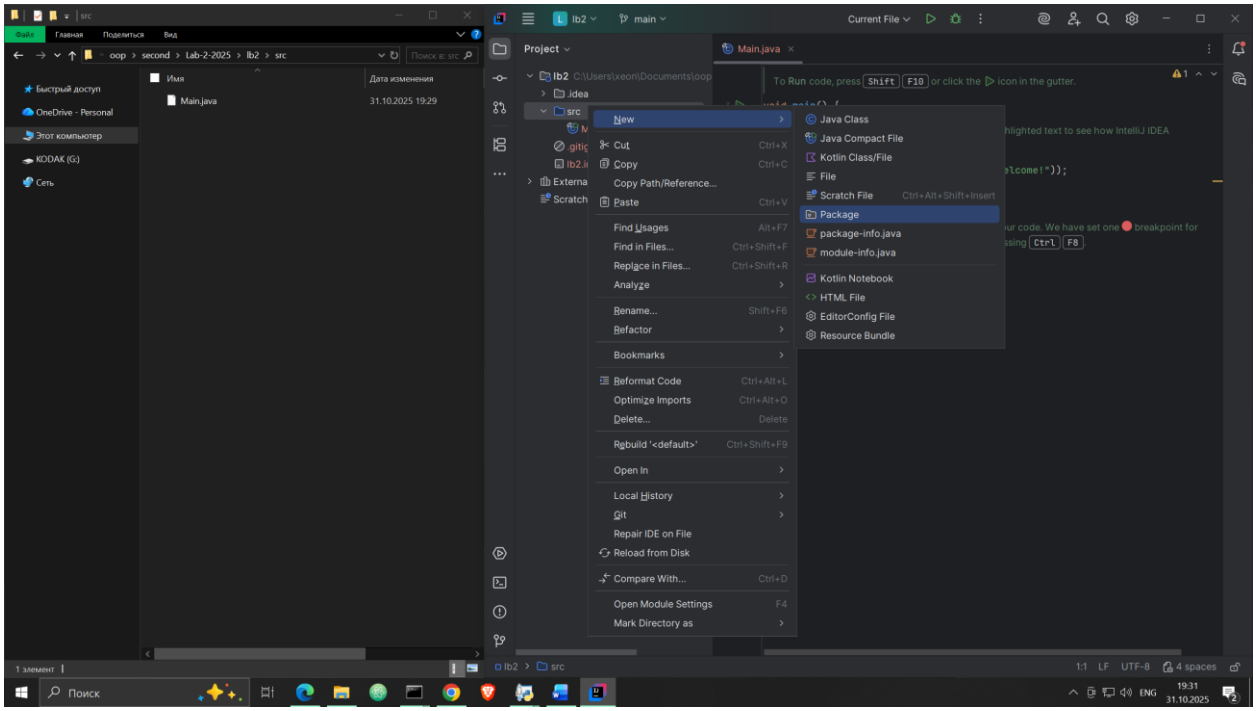
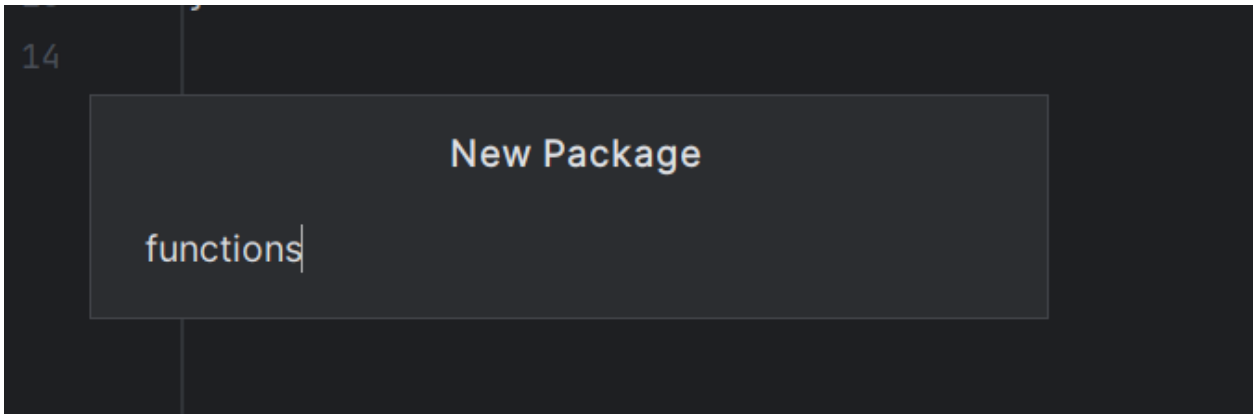
Отчёт по лабораторной работе №2

Тенигин Валерий 6204-010302D

Задание 1

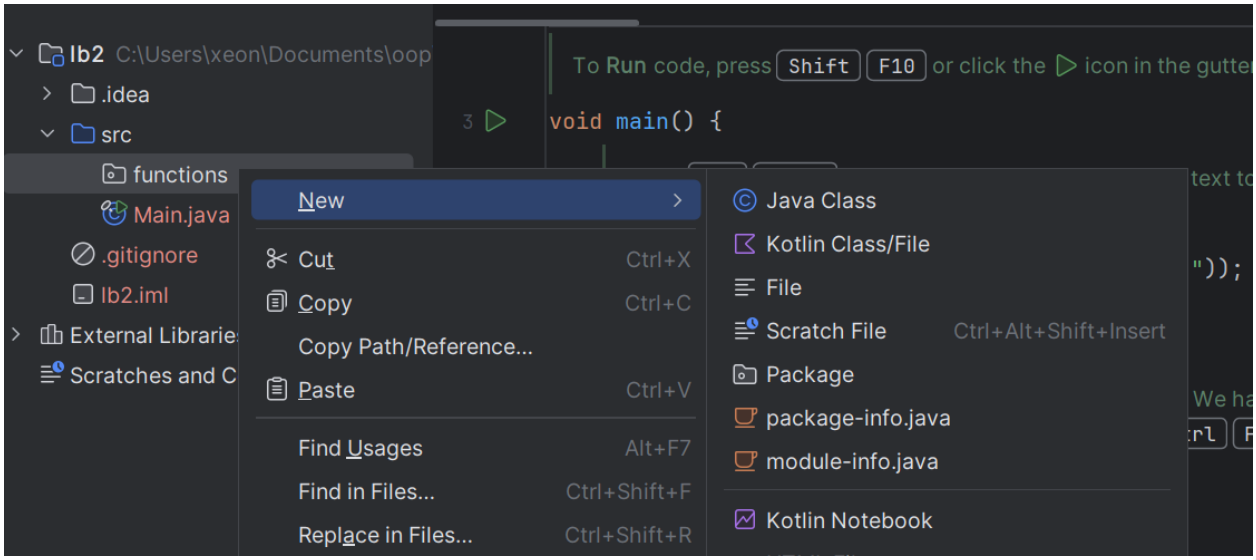
Создание пакета functions

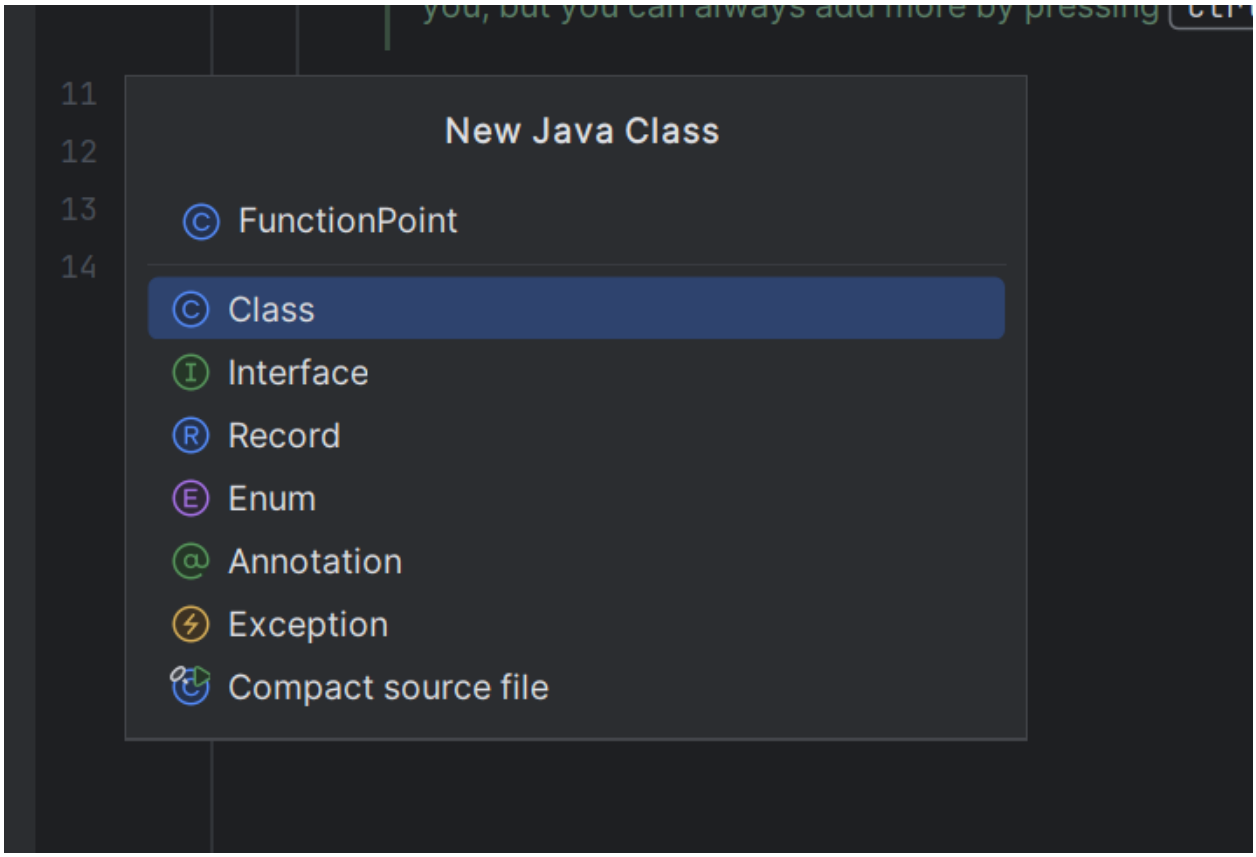


Задание 2

Внутри созданного пакета создаём класс FunctionPoint



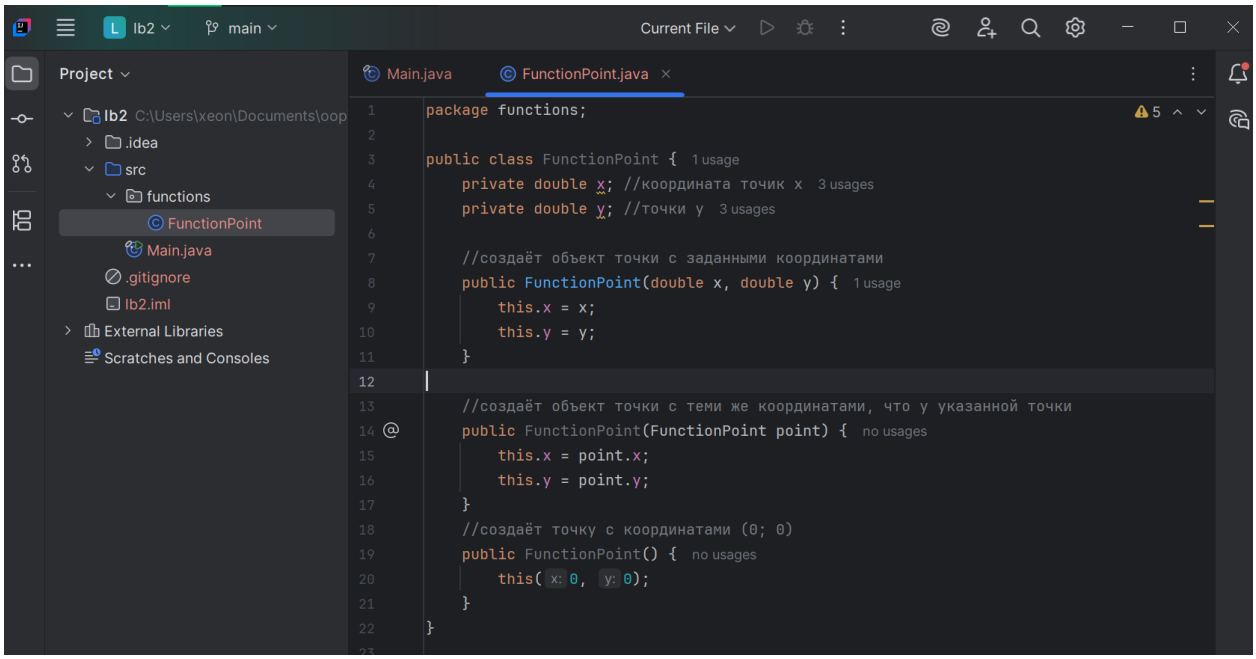




В этом классе описываем следующие конструкторы:

• FunctionPoint(double x, double y) – создаёт объект точки с заданными координатами;

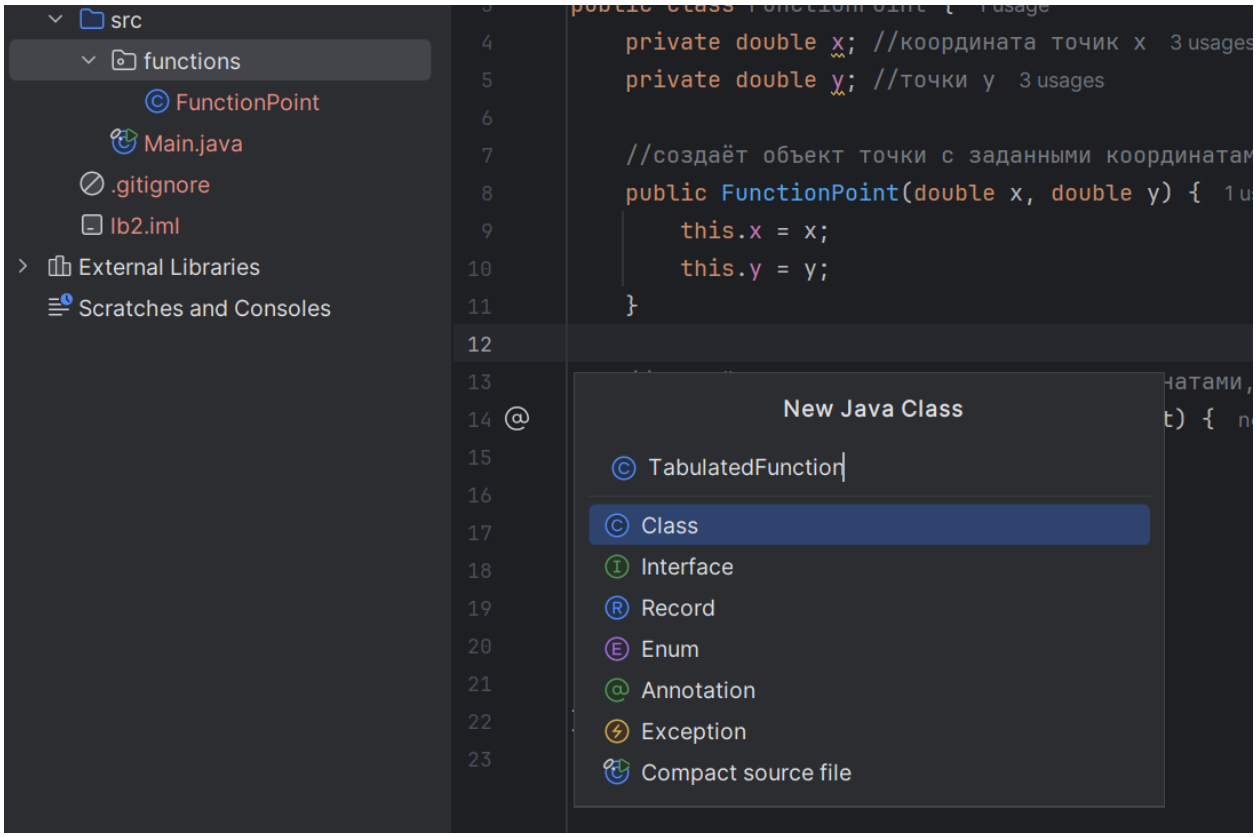
• FunctionPoint(FunctionPoint point) – создаёт объект точки с теми же координатами, что у указанной точки;

• FunctionPoint() – создаёт точку с координатами (0; 0).




Задание 3

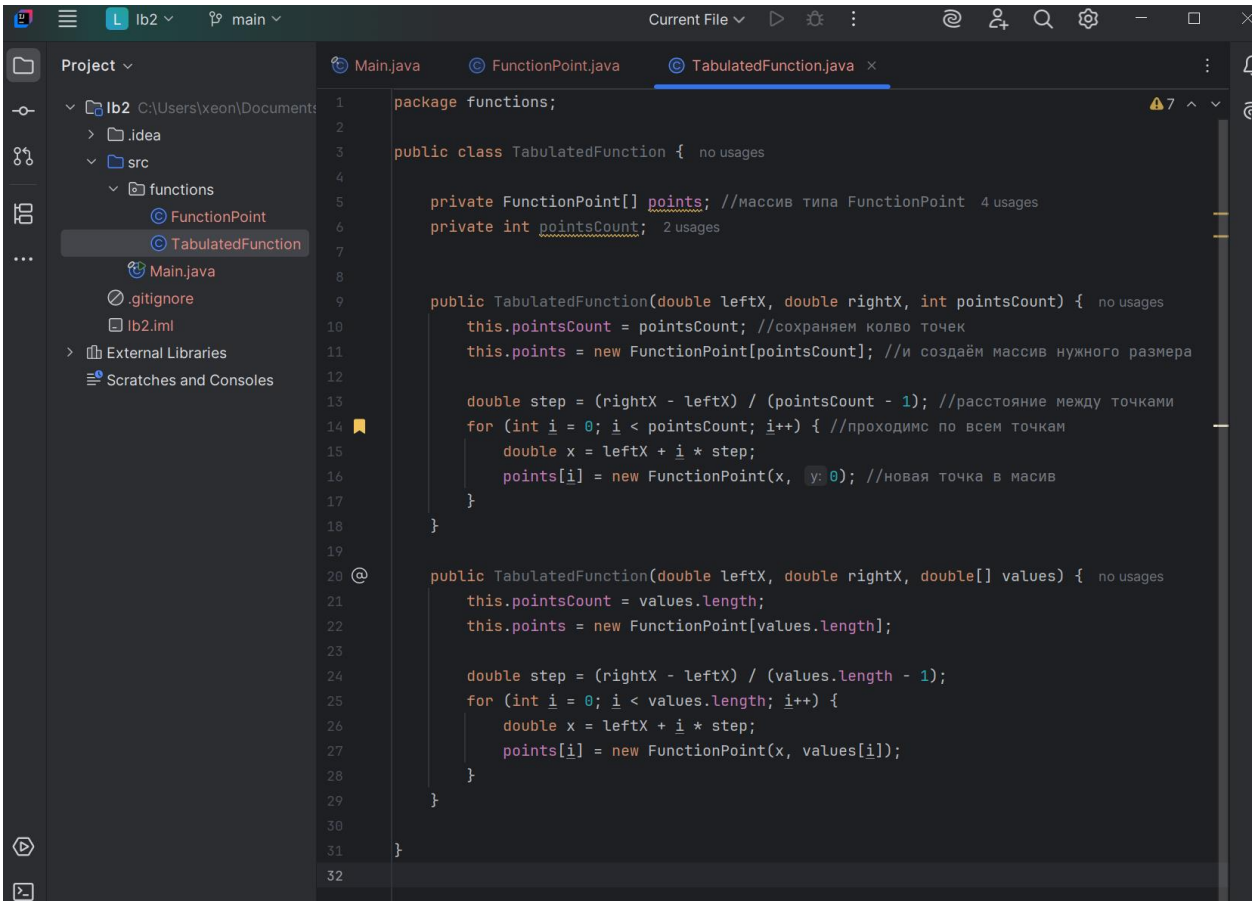
Создаём ещё один класс TabulatedFunction



Создаём следующие конструкторы:

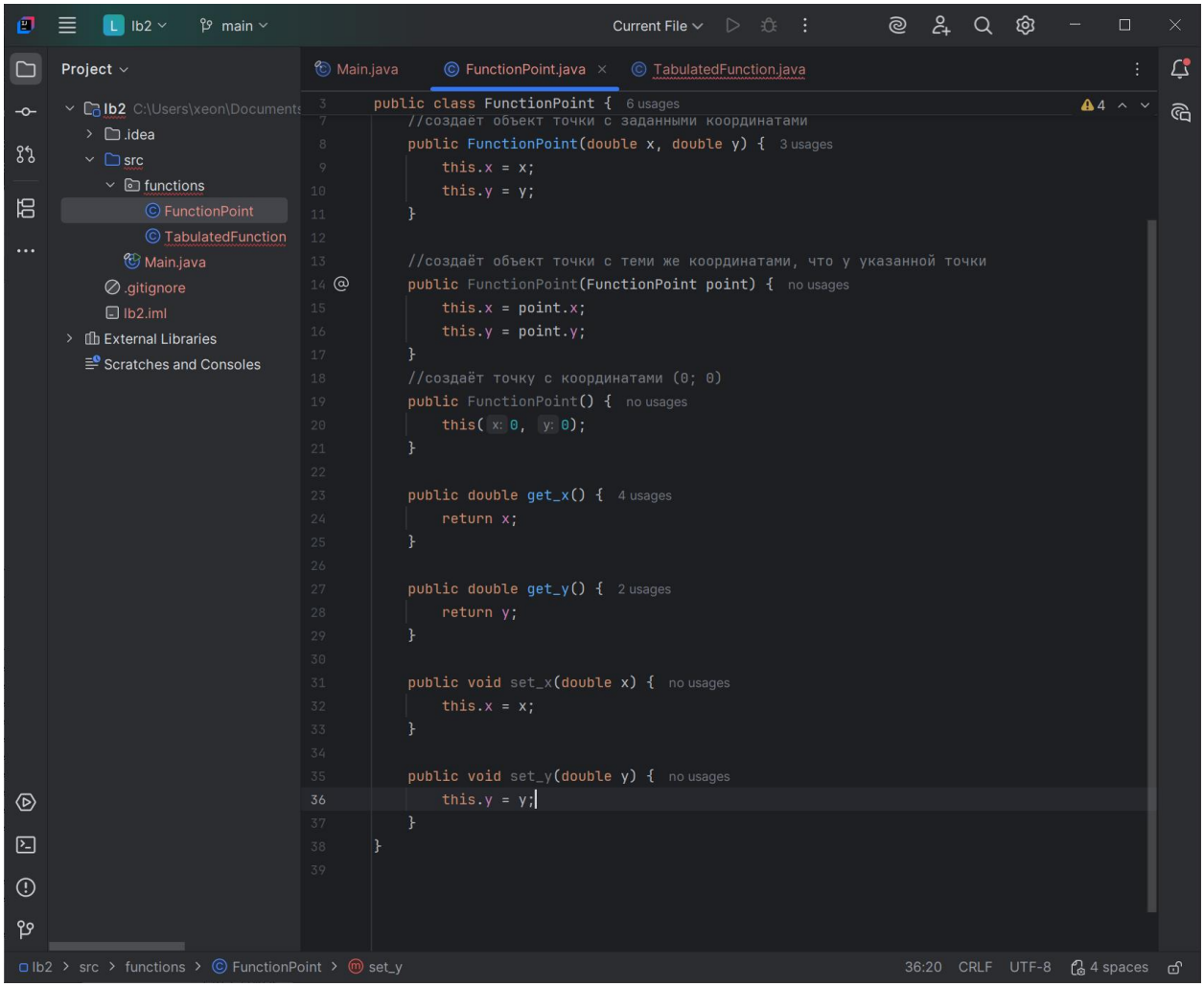
• TabulatedFunction(double leftX, double rightX, int pointsCount) – создаёт объект табулированной функции по заданным левой и правой границе области определения, а также количеству точек для табулирования (значения функции в точках при этом следует считать равными 0);

• TabulatedFunction(double leftX, double rightX, double[] values) – аналогичен предыдущему конструктору, но вместо количества точек получает значения функции в виде массива.



Задание 4

Добавил гетеры и сетеры в класс FunctionPoint



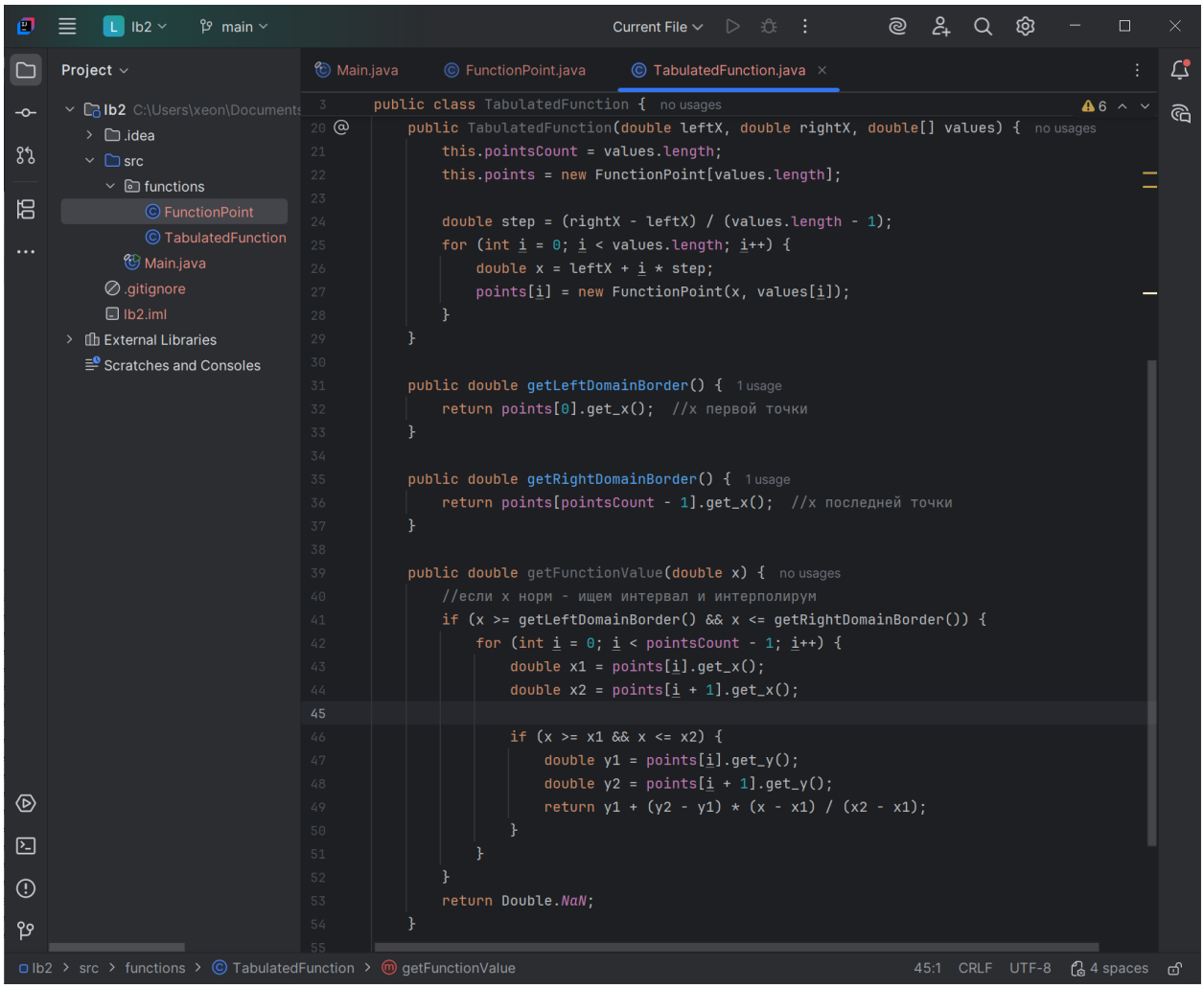

Длюавлены следующие методы:

• Метод double getLeftDomainBorder() должен возвращать значение левой границы области определения табулированной функции.

• метод double getRightDomainBorder() должен возвращать значение правой границы области определения табулированной функции.

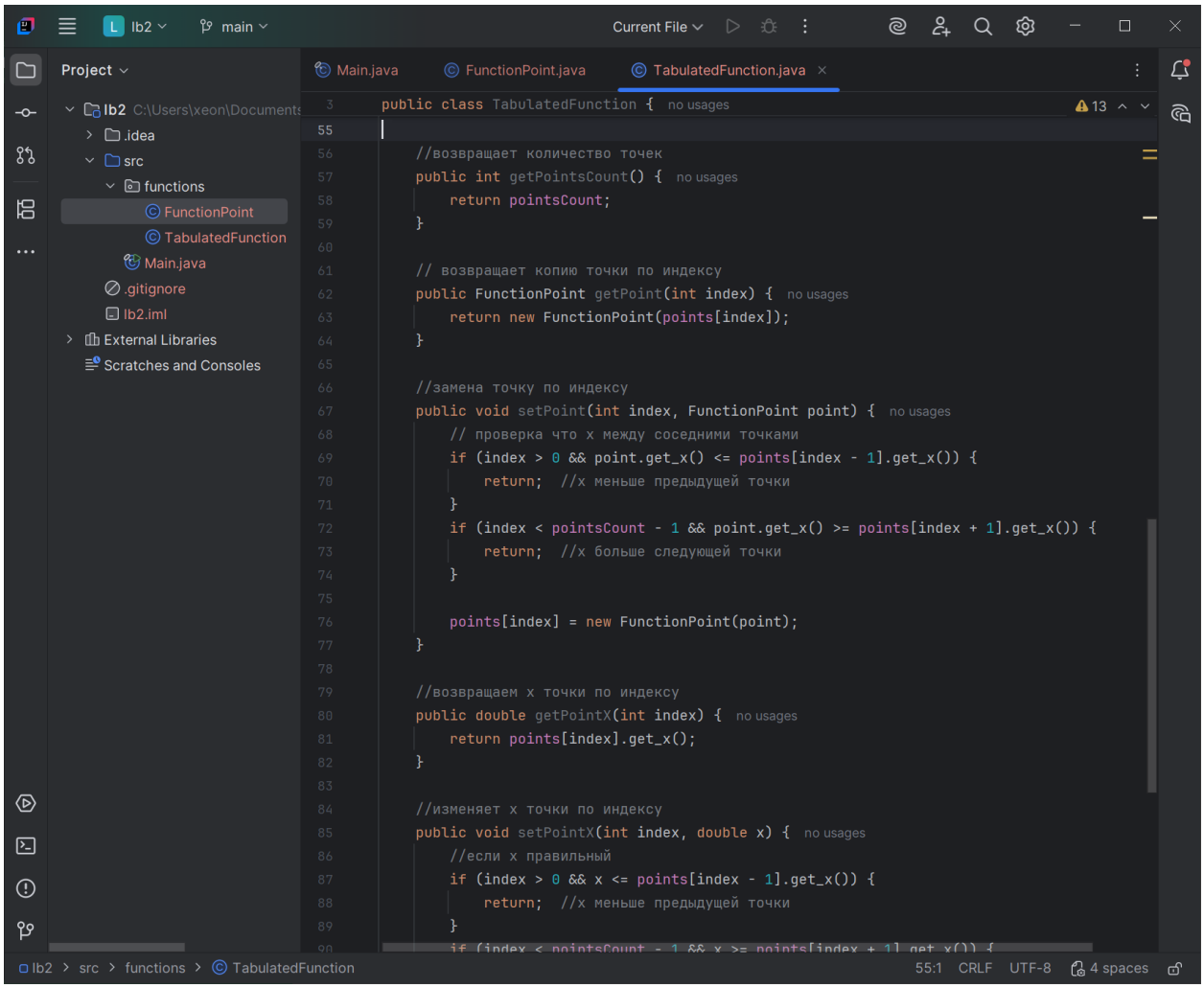
• Метод double getFunctionValue(double x) должен возвращать значение функции в точке x, если эта точка лежит в области

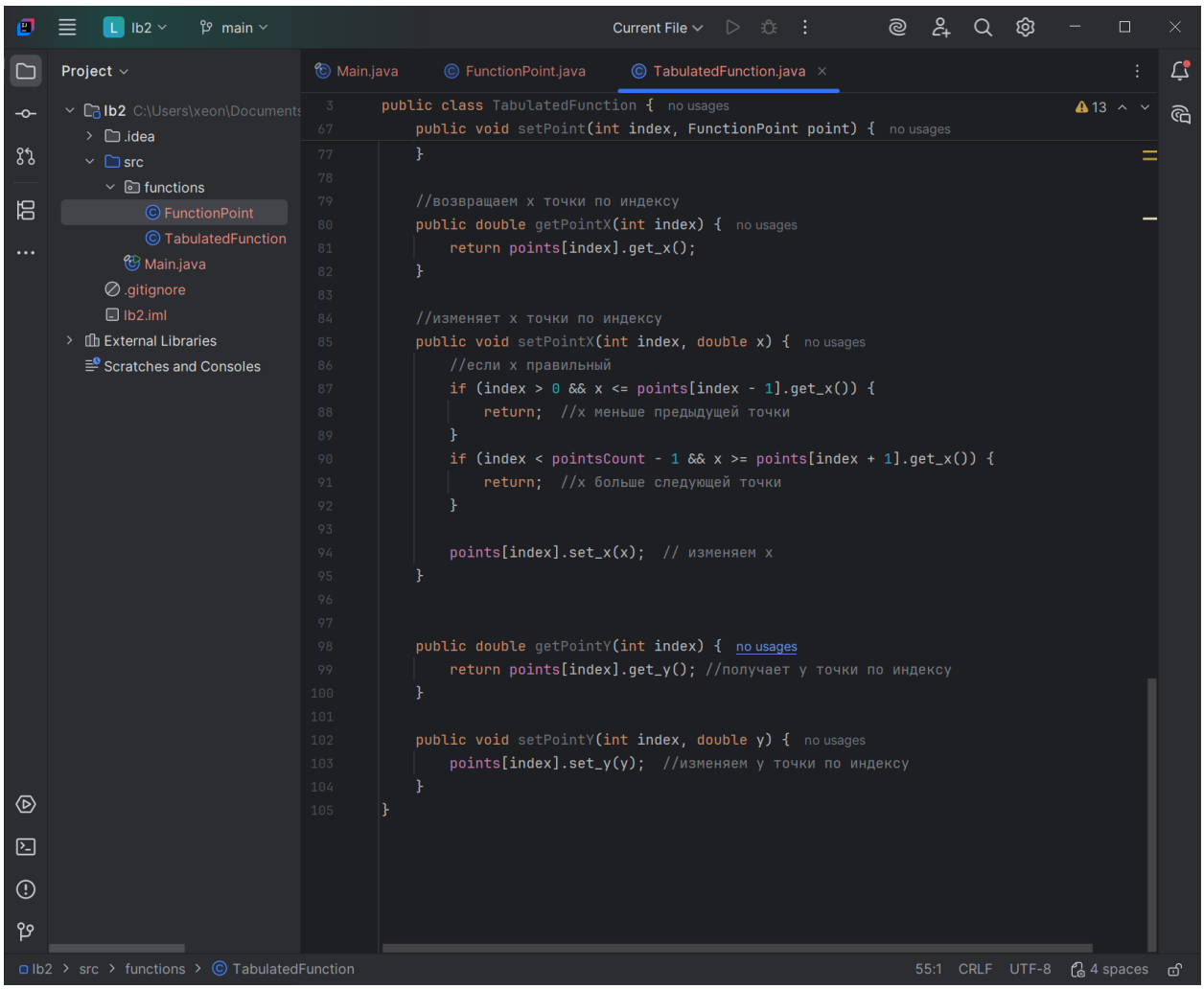
определения функции



Задание 5

Добавлены методы для работы с точками таблуированной функции согласно заданию





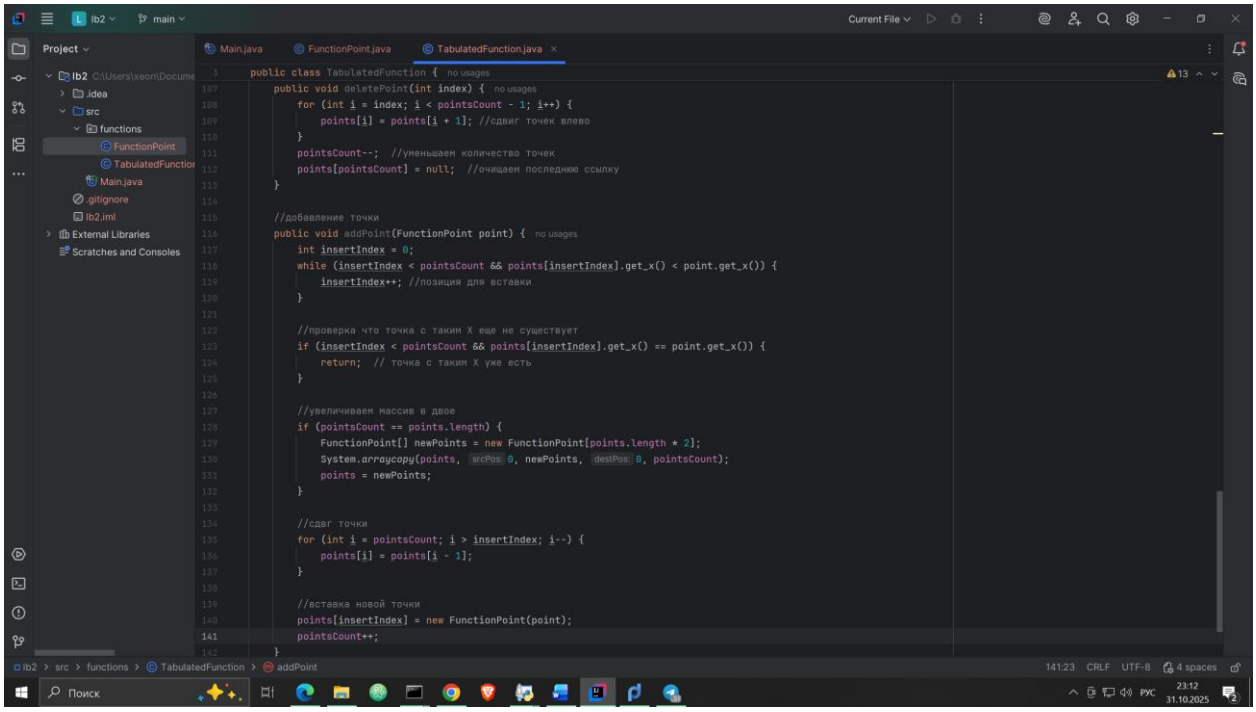

```
3   public class TabulatedFunction {  no usages
67      public void setPoint(int index, FunctionPoint point) {  no usages
77      }
78
79      //возвращаем x точки по индексу
80      public double getPointX(int index) {  no usages
81          return points[index].get_x();
82      }
83
84      //изменяет x точки по индексу
85      public void setPointX(int index, double x) {  no usages
86          //если x правильный
87          if (index > 0 && x <= points[index - 1].get_x()) {
88              return;  //x меньше предыдущей точки
89          }
90          if (index < pointsCount - 1 && x >= points[index + 1].get_x()) {
91              return;  //x больше следующей точки
92          }
93
94          points[index].set_x(x);  // изменяем x
95      }
96
97
98      public double getPointY(int index) {  no usages
99          return points[index].get_y(); //получает y точки по индексу
100     }
101
102     public void setPointY(int index, double y) {  no usages
103         points[index].set_y(y);  //изменяем y точки по индексу
104     }
105 }
```

Задание 6

Добавлены методы изменяющие количество точек таблуированной функции:

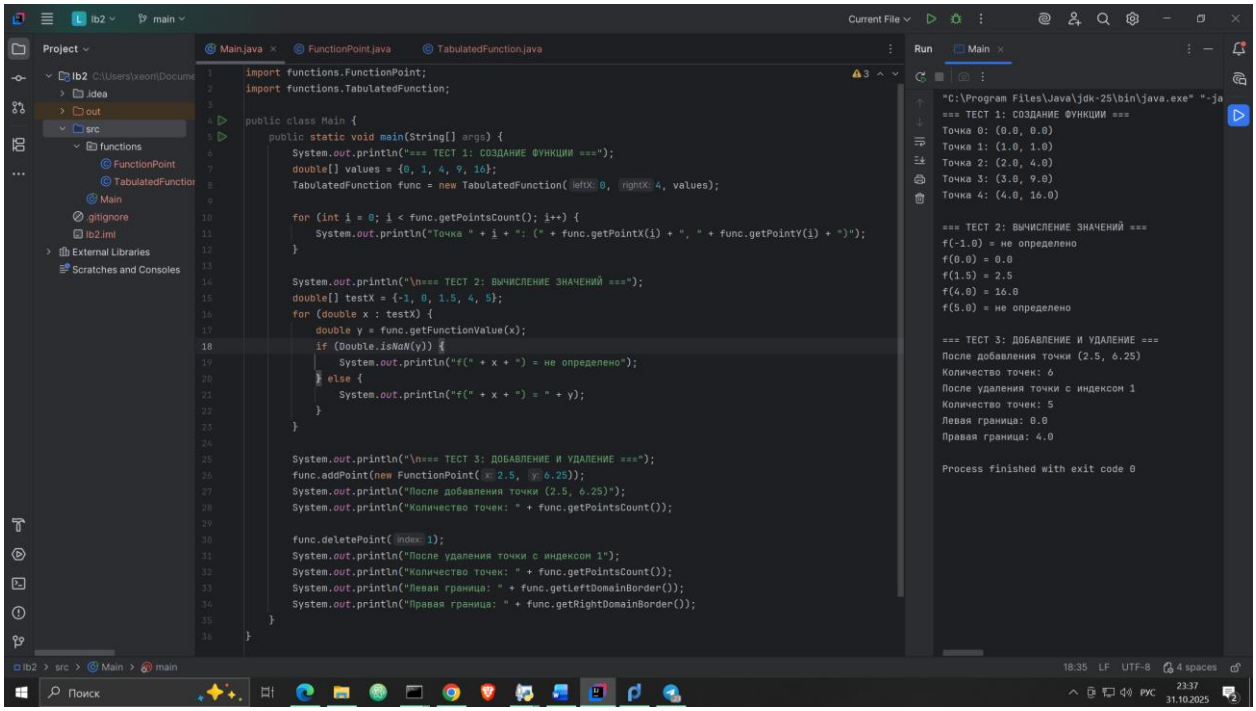
• Метод void deletePoint(int index) должен удалять заданную точку табулированной функции.

• Метод void addPoint(FunctionPoint point) должен добавлять новую точку табулированной функции.



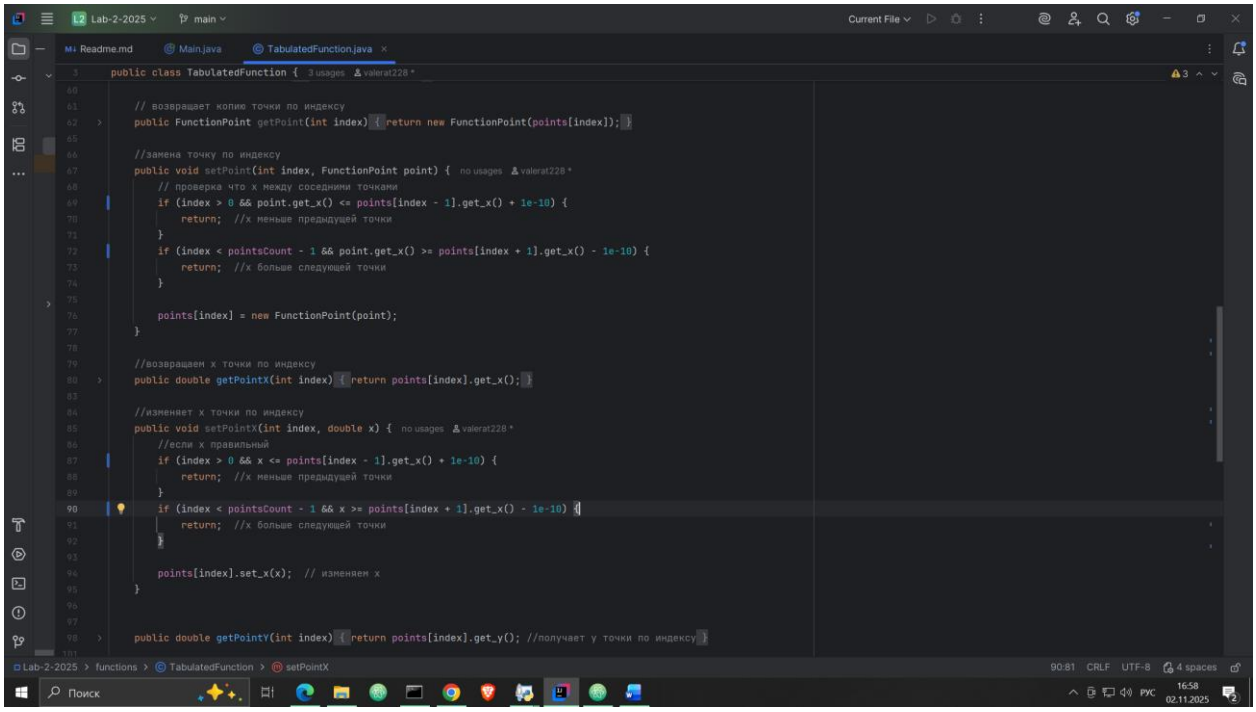
Задание 7

Проверка работы

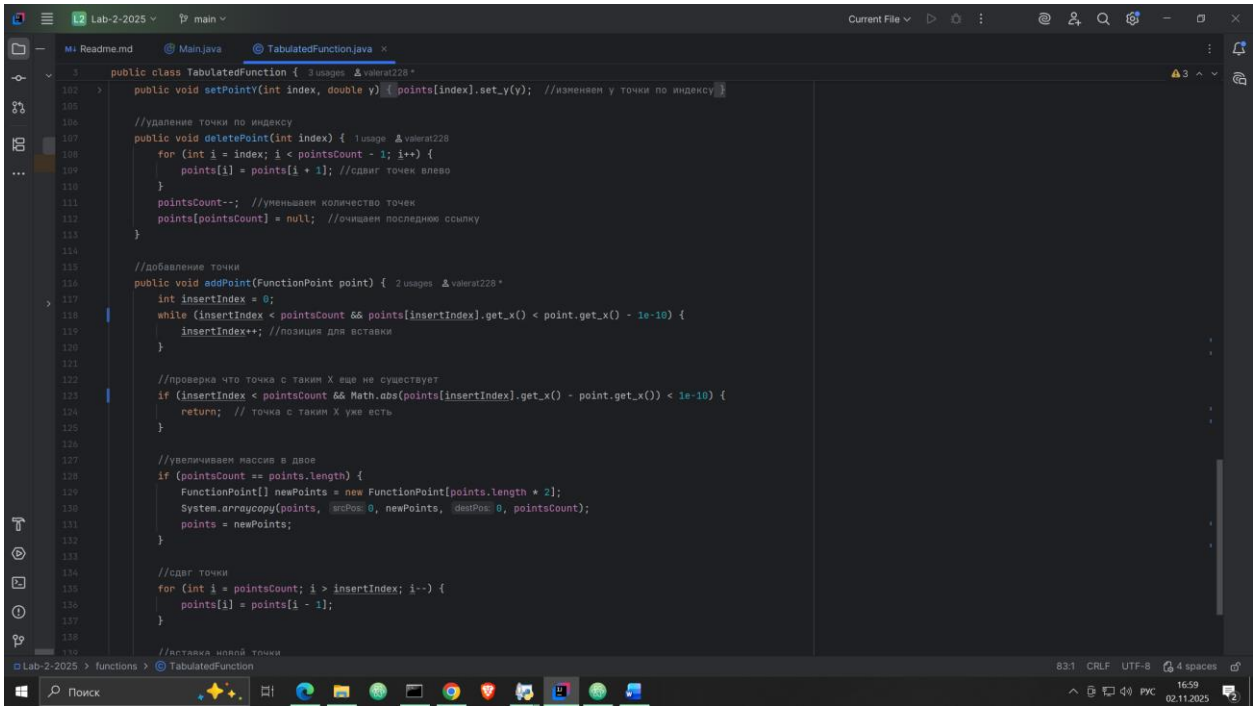


Доработанный код 1

Стал использовать эпсилон при сравнении

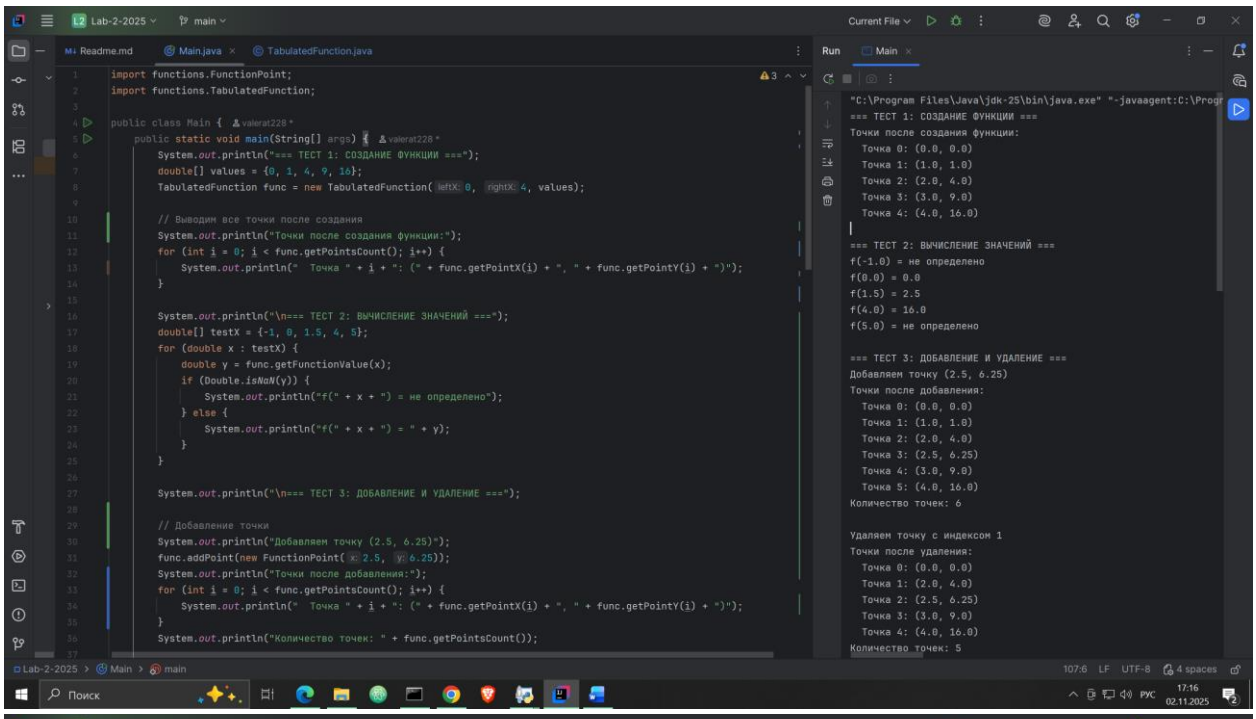


```
public class TabulatedFunction {  3 usages  valerat228 *

    // возвращает копию точки по индексу
    public FunctionPoint getPoint(int index) { return new FunctionPoint(points[index]); }

    //замена точку по индексу
    public void setPoint(int index, FunctionPoint point) {  no usages  valerat228 *
        // проверка что х между соседними точками
        if (index > 0 && point.get_x() <= points[index - 1].get_x() + 1e-10) {
            return;  //х меньше предыдущей точки
        }
        if (index < pointsCount - 1 && point.get_x() >= points[index + 1].get_x() - 1e-10) {
            return;  //х больше следующей точки
        }

        points[index] = new FunctionPoint(point);
    }

    //возвращаем х точки по индексу
    public double getPointX(int index) { return points[index].get_x(); }

    //изменяет х точки по индексу
    public void setPointX(int index, double x) {  no usages  valerat228 *
        //если х правильный
        if (index > 0 && x <= points[index - 1].get_x() + 1e-10) {
            return;  //х меньше предыдущей точки
        }
        if (index < pointsCount - 1 && x >= points[index + 1].get_x() - 1e-10) {
            return;  //х больше следующей точки
        }

        points[index].set_x(x);  // изменяем х
    }


    public double getPointY(int index) { return points[index].get_y(); //получает у точки по индексу }
```



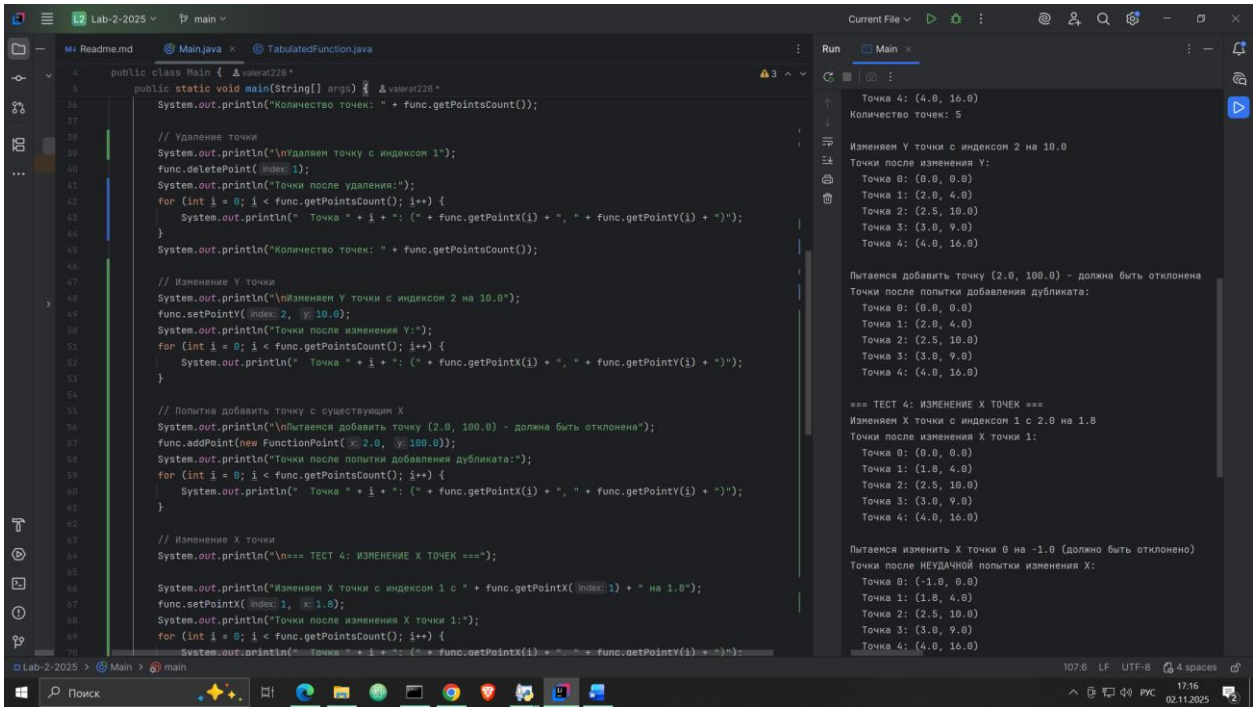
```
public class TabulatedFunction {  3 usages  valerat228 *
    public void setPointY(int index, double y) { points[index].set_y(y);  //изменяем у точки по индексу }

    //удаление точки по индексу
    public void deletePoint(int index) {  1 usage  valerat228
        for (int i = index; i < pointsCount - 1; i++) {
            points[i] = points[i + 1]; //сдвиг точек влево
        }
        pointsCount--;  //уменьшаем количество точек
        points[pointsCount] = null;  //очищаем последнюю ссылку
    }

    //добавление точки
    public void addPoint(FunctionPoint point) {  2 usages  valerat228 *
        int insertIndex = 0;
        while (insertIndex < pointsCount && points[insertIndex].get_x() < point.get_x() - 1e-10) {
            insertIndex++; //позиция для вставки
        }

        //проверка что точка с таким X еще не существует
        if (insertIndex < pointsCount && Math.abs(points[insertIndex].get_x() - point.get_x()) < 1e-10) {
            return;  // точка с таким X уже есть
        }

        //увеличиваем массив в двое
        if (pointsCount == points.length) {
            FunctionPoint[] newPoints = new FunctionPoint[points.length * 2];
            System.arraycopy(points,  srcPos: 0, newPoints,  destPos: 0, pointsCount);
            points = newPoints;
        }

        //сдвг точки
        for (int i = pointsCount; i > insertIndex; i--) {
            points[i] = points[i - 1];
        }
```

Изменил проверку работы



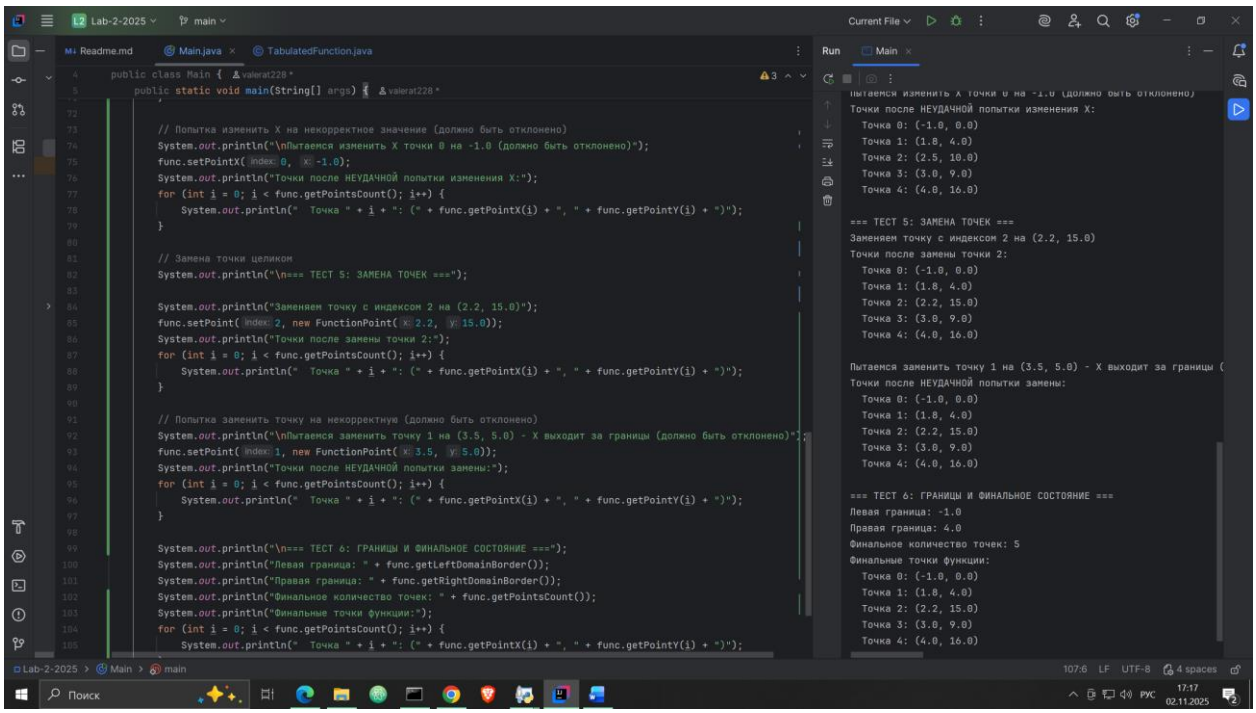
```java
import functions.FunctionPoint;
import functions.TabulatedFunction;

public class Main {
    public static void main(String[] args) {
        System.out.println("=== ТЕСТ 1: СОЗДАНИЕ ФУНКЦИИ ===");
        double[] values = {0, 1, 4, 9, 16};
        TabulatedFunction func = new TabulatedFunction( leftX: 0,  rightX: 4, values);

        // Выводим все точки после создания
        System.out.println("Точки после создания функции:");
        for (int i = 0; i < func.getPointsCount(); i++) {
            System.out.println("  Точка " + i + ": (" + func.getPointX(i) + ", " + func.getPointY(i) + ")");
        }

        System.out.println("\n=== ТЕСТ 2: ВЫЧИСЛЕНИЕ ЗНАЧЕНИЙ ===");
        double[] testX = {-1, 0, 1.5, 4, 5};
        for (double x : testX) {
            double y = func.getFunctionValue(x);
            if (Double.isNaN(y)) {
                System.out.println("f(" + x + ") = не определено");
            } else {
                System.out.println("f(" + x + ") = " + y);
            }
        }

        System.out.println("\n=== ТЕСТ 3: ДОБАВЛЕНИЕ И УДАЛЕНИЕ ===");

        // Добавление точки
        System.out.println("Добавляем точку (2.5, 6.25)");
        func.addPoint(new FunctionPoint( x: 2.5,  y: 6.25));
        System.out.println("Точки после добавления:");
        for (int i = 0; i < func.getPointsCount(); i++) {
            System.out.println("  Точка " + i + ": (" + func.getPointX(i) + ", " + func.getPointY(i) + ")");
        }
        System.out.println("Количество точек: " + func.getPointsCount());
```

```
"C:\Program Files\Java\jdk-25\bin\java.exe" "-javaagent:C:\Progr
=== ТЕСТ 1: СОЗДАНИЕ ФУНКЦИИ ===
Точки после создания функции:
  Точка 0: (0.0, 0.0)
  Точка 1: (1.0, 1.0)
  Точка 2: (2.0, 4.0)
  Точка 3: (3.0, 9.0)
  Точка 4: (4.0, 16.0)

=== ТЕСТ 2: ВЫЧИСЛЕНИЕ ЗНАЧЕНИЙ ===
f(-1.0) = не определено
f(0.0) = 0.0
f(1.5) = 2.5
f(4.0) = 16.0
f(5.0) = не определено

=== ТЕСТ 3: ДОБАВЛЕНИЕ И УДАЛЕНИЕ ===
Добавляем точку (2.5, 6.25)
Точки после добавления:
  Точка 0: (0.0, 0.0)
  Точка 1: (1.0, 1.0)
  Точка 2: (2.0, 4.0)
  Точка 3: (2.5, 6.25)
  Точка 4: (3.0, 9.0)
  Точка 5: (4.0, 16.0)
Количество точек: 6

Удаляем точку с индексом 1
Точки после удаления:
  Точка 0: (0.0, 0.0)
  Точка 1: (2.0, 4.0)
  Точка 2: (2.5, 6.25)
  Точка 3: (3.0, 9.0)
  Точка 4: (4.0, 16.0)
Количество точек: 5
```



```java
        System.out.println("Количество точек: " + func.getPointsCount());

        // Удаление точки
        System.out.println("\nУдаляем точку с индексом 1");
        func.deletePoint( index: 1);
        System.out.println("Точки после удаления:");
        for (int i = 0; i < func.getPointsCount(); i++) {
            System.out.println("  Точка " + i + ": (" + func.getPointX(i) + ", " + func.getPointY(i) + ")");
        }
        System.out.println("Количество точек: " + func.getPointsCount());

        // Изменение Y точки
        System.out.println("\nИзменяем Y точки с индексом 2 на 10.0");
        func.setPointY( index: 2,  y: 10.0);
        System.out.println("Точки после изменения Y:");
        for (int i = 0; i < func.getPointsCount(); i++) {
            System.out.println("  Точка " + i + ": (" + func.getPointX(i) + ", " + func.getPointY(i) + ")");
        }

        // Попытка добавить точку с существующим X
        System.out.println("\nПытаемся добавить точку (2.0, 100.0) - должна быть отклонена");
        func.addPoint(new FunctionPoint( x: 2.0,  y: 100.0));
        System.out.println("Точки после попытки добавления дубликата:");
        for (int i = 0; i < func.getPointsCount(); i++) {
            System.out.println("  Точка " + i + ": (" + func.getPointX(i) + ", " + func.getPointY(i) + ")");
        }

        // Изменение X точки
        System.out.println("\n=== ТЕСТ 4: ИЗМЕНЕНИЕ X ТОЧЕК ===");

        System.out.println("Изменяем X точки с индексом 1 с " + func.getPointX( index: 1) + " на 1.8");
        func.setPointX( index: 1,  x: 1.8);
        System.out.println("Точки после изменения X точки 1:");
        for (int i = 0; i < func.getPointsCount(); i++) {
            System.out.println("  Точка " + i + ": (" + func.getPointX(i) + ", " + func.getPointY(i) + ")");
```

```
  Точка 4: (4.0, 16.0)
Количество точек: 5

Изменяем Y точки с индексом 2 на 10.0
Точки после изменения Y:
  Точка 0: (0.0, 0.0)
  Точка 1: (2.0, 4.0)
  Точка 2: (2.5, 10.0)
  Точка 3: (3.0, 9.0)
  Точка 4: (4.0, 16.0)

Пытаемся добавить точку (2.0, 100.0) - должна быть отклонена
Точки после попытки добавления дубликата:
  Точка 0: (0.0, 0.0)
  Точка 1: (2.0, 4.0)
  Точка 2: (2.5, 10.0)
  Точка 3: (3.0, 9.0)
  Точка 4: (4.0, 16.0)

=== ТЕСТ 4: ИЗМЕНЕНИЕ X ТОЧЕК ===
Изменяем X точки с индексом 1 с 2.0 на 1.8
Точки после изменения X точки 1:
  Точка 0: (0.0, 0.0)
  Точка 1: (1.8, 4.0)
  Точка 2: (2.5, 10.0)
  Точка 3: (3.0, 9.0)
  Точка 4: (4.0, 16.0)

Пытаемся изменить X точки 0 на -1.0 (должно быть отклонено)
Точки после НЕУДАЧНОЙ попытки изменения X:
  Точка 0: (-1.0, 0.0)
  Точка 1: (1.8, 4.0)
  Точка 2: (2.5, 10.0)
  Точка 3: (3.0, 9.0)
  Точка 4: (4.0, 16.0)
```




Найдена ошибка

```
Пытаемся изменить X точки 0 на -1.0 (должно быть отклонено)
Точки после НЕУДАЧНОЙ попытки изменения X:
  Точка 0: (-1.0, 0.0)
  Точка 1: (1.8, 4.0)
  Точка 2: (2.5, 10.0)
  Точка 3: (3.0, 9.0)
  Точка 4: (4.0, 16.0)
```

# Изменённый код для устранения ошибки