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

Отчет по лабораторной работе №3

Дисциплина: «ООП»

Тема «Исключения и интерфейсы»

Выполнил: Пантелеев Ю.В.

Группа: 6201-120303

Самара, 2025

Задание на лабораторную работу

Задание 1

Я ознакомился со следующими классами исключений:

java.lang.Exception - базовый класс для всех проверяемых исключений, которые можно обработать.

java.lang.IndexOutOfBoundsException - выход индекса за границы коллекции или массива.

java.lang.ArrayIndexOutOfBoundsException - конкретно для выхода за границы массива.

java.lang.IllegalArgumentException - недопустимый аргумент метода.

java.lang.IllegalStateException - вызов метода в некорректном состоянии объекта.

Задание 2

В пакете functions я создал два класса исключений:

FunctionPointIndexOutOfBoundsException

```java
package functions;

public class FunctionPointIndexOutOfBoundsException extends IndexOutOfBoundsException {  21 usages  new *
    public FunctionPointIndexOutOfBoundsException(String message) {  14 usages  new *
        super(message);
    }
}
```

InappropriateFunctionPointException

```java
package functions;

public class InappropriateFunctionPointException extends Exception {  28 usages  new *
    public InappropriateFunctionPointException(String message) { super(message); }
}

```

Задание 3

В разработанный ранее класс TabulatedFunction я внес изменения, обеспечивающие выбрасывание исключений методами класса.

Оба конструктора класса:

```
public ArrayTabulatedFunction(double leftX, double rightX, int pointsCount){  1 usage  new *
    if(leftX - rightX > - EPSILON){
        throw new IllegalArgumentException("Левая граница диапазона больше или равна правой");
    }
    else if (pointsCount < 2) {
        throw new IllegalArgumentException("Количество предлагаемых точек меньше двух");
    }
    this.pointsCount = pointsCount;
    this.points = new FunctionPoint[pointsCount];

    double step = (rightX - leftX)/(pointsCount - 1);
    for (int i =0; i < pointsCount; i++){
        points[i]=new FunctionPoint( x: leftX + i * step, y: 0);
    }
}
```

```
public ArrayTabulatedFunction(double leftX, double rightX, double[] values){  no usages  new *
    if(leftX - rightX > - EPSILON){
        throw new IllegalArgumentException("Левая граница диапазона больше или равна правой");
    }
    else if (values.length<2) {
        throw new IllegalArgumentException("Количество предлагаемых точек меньше двух");
    }

    this.pointsCount = values.length;
    this.points = new FunctionPoint[pointsCount];

    double step = (rightX-leftX)/(pointsCount-1);
    for (int i =0; i<pointsCount; i++){
        points[i]=new FunctionPoint( x: leftX+i*step,values[i]);
    }
}
```

Методы:

getPoint()

```
public FunctionPoint getPoint(int index){  1 usage  new *
    if (index<0||index>=pointsCount){
        throw new FunctionPointIndexOutOfBoundsException("Введённый индекс выходит за границы набора точек или меньше нуля");
    }
    return new FunctionPoint(points[index].getX(),points[index].getY());
}
```

setPoint()

```
public void setPoint(int index, FunctionPoint point) throws InappropriateFunctionPointException {  3 usages  new *
    if (index<0||index>=pointsCount){
        throw new FunctionPointIndexOutOfBoundsException("Введённый индекс выходит за границы набора точек или меньше нуля");
    }
    else if(index == 0){
        if(point.getX() - points[1].getX() > -EPSILON){
            throw new InappropriateFunctionPointException("Координата x задаваемой точки лежит вне интервала");
        }
    }
    else if(index == pointsCount-1){
        if(point.getX() - points[pointsCount-2].getX() < EPSILON){
            throw new InappropriateFunctionPointException("Координата x задаваемой точки лежит вне интервала");
        }
    }
    else if (points[index-1].getX() - point.getX() > -EPSILON || points[index + 1].getX() - point.getX() < EPSILON) {
        throw new InappropriateFunctionPointException("Координата x задаваемой точки лежит вне интервала");
    }
    points[index]= new FunctionPoint(point.getX(),point.getY());
}
```

getPointX()

```
public double getPointX(int index){  18 usages  new *
    if (index<0||index>=pointsCount){
        throw new FunctionPointIndexOutOfBoundsException("Введённый индекс выходит за границы набора точек или меньше нуля");
    }
    return points[index].getX();
}
```

setPointX()

```
public void setPointX(int index, double x) throws InappropriateFunctionPointException{  3 usages  new *
    if (index < 0 || index >= pointsCount) {
        throw new FunctionPointIndexOutOfBoundsException("Введённый индекс выходит за границы набора точек или меньше нуля");
    }
    else if(index == 0){
        if(x - points[1].getX() > -EPSILON){
            throw new InappropriateFunctionPointException("Координата x задаваемой точки лежит вне интервала");
        }
    }
    else if(index == pointsCount-1){
        if(x - points[pointsCount-2].getX() < EPSILON){
            throw new InappropriateFunctionPointException("Координата x задаваемой точки лежит вне интервала");
        }
    }
    else if (points[index - 1].getX() - x >-EPSILON|| x - points[index + 1].getX() < EPSILON){
        throw new InappropriateFunctionPointException("Координата x задаваемой точки лежит вне интервала");
    }
    points[index].setX(x);
}
```

getPointY()

```
public double getPointY(int index){  16 usages  new *
    if (index<0||index>=pointsCount){
        throw new FunctionPointIndexOutOfBoundsException("Введённый индекс выходит за границы набора точек или меньше нуля");
    }
    return points[index].getY();
}
```

setPointY()

```
public void setPointY(int index, double y){  3 usages  new *
    if (index<0||index>=pointsCount){
        throw new FunctionPointIndexOutOfBoundsException("Введённый индекс выходит за границы набора точек или меньше нуля");
    }
    points[index].setY(y);
}
```

deletePoint

```
public void deletePoint(int index){  2 usages  new *
    if(pointsCount<3){
        throw new IllegalStateException("В массиве находятся только две точки");
    }
    if (index<0||index>=pointsCount){
        throw new FunctionPointIndexOutOfBoundsException("Введённый индекс выходит за границы набора точек или меньше нуля");
    }
    System.arraycopy(points,index,points, destPos: index-1, length: pointsCount-index);
    points[pointsCount-1]=null;
    pointsCount--;
}
```

addPoint

```
public void addPoint(FunctionPoint point) throws InappropriateFunctionPointException{  3 usages  new *
    for (int i = 1; i < pointsCount; i++) {
        if (point.getX() - points[i].getX() < -EPSILON && point.getX() - points[i - 1].getX() > EPSILON) {
            FunctionPoint[] newPoints = new FunctionPoint[pointsCount + 1];
            System.arraycopy(points,  srcPos: 0, newPoints,  destPos: 0, i);
            newPoints[i] = point;
            System.arraycopy(points, i, newPoints,  destPos: i + 1,  length: pointsCount - i);
            points = newPoints;
            i = pointsCount;
            pointsCount+=1;
        }
        else if(Math.abs(point.getX()-points[i].getX())<EPSILON||Math.abs(point.getX()-points[i-1].getX())<EPSILON){
            throw new InappropriateFunctionPointException("В наборе точек функции есть точка, абсцисса которой совпадает с абсциссой добавляемой точки");
        }
    }
}
```

Задание 4

4.1 Я описал класс элементов списка FunctionNode

```
private static class FunctionNode {  21 usages  new *
    FunctionPoint point;  31 usages
    FunctionNode previous;  15 usages
    FunctionNode next;  22 usages

    FunctionNode(FunctionPoint point, FunctionNode prev, FunctionNode next) {  2 usages  new *
        this.point = point;
        this.previous = prev;
        this.next = next;
    }
    FunctionNode(FunctionPoint point) {  1 usage  new *
        this.point = point;
        this.previous = null;
        this.next = null;
    }
}
```

Описал я его внутри класса LinkedListTabulatedFunction, так как:

1. FunctionNode является внутренней деталью реализации связного списка.
2. Не имеет смысла вне контекста LinkedListTabulatedFunction

Видимость: private static:

1. private – полная инкапсуляция, недоступен извне класса.
2. static – не хранит ссылку на внешний класс, экономит память.

4.2. Далее в классе LinkedListTabulatedFunction я описал методы для работы с объектами этого класса:

```
package functions;

public class LinkedListTabulatedFunction implements TabulatedFunction{  4 usages  new *
    public static final double EPSILON = 1e-10;  15 usages
    private FunctionNode head;  22 usages
    private int pointsCount;  22 usages
    private FunctionNode lastNode;  6 usages
    private int lastIndex;  11 usages
```

FunctionNode getNodeByIndex(int index), возвращающий ссылку на объект элемента списка по его номеру.

```
    private FunctionNode getNodeByIndex(int index){  16 usages  new *
        FunctionNode node;
        int startIndex;

        if(lastIndex!=-1&& Math.abs(index-lastIndex)<Math.abs(index)) {
            node = lastNode;
            startIndex=lastIndex;
        }
        else{
            node=head.next;
            startIndex=0;
        }

        if(index<=startIndex){
            for(int i = startIndex; i>index; i--){
                node = node.previous;
            }
        }
        else{
            for(int i = startIndex; i<index; i++){
                node = node.next;
            }
        }
        lastNode = node;
        lastIndex = index;
        return node;
    }
```

FunctionNode addNodeToTail(), добавляющий новый элемент в конец списка и возвращающий ссылку на объект этого элемента.

```
private FunctionNode addNodeToTail(){  2 usages  new *
    FunctionNode newNode = new FunctionNode( point: null,head.previous,head);
    head.previous.next = newNode;
    head.previous=newNode;
    pointsCount++;
    lastNode = newNode;
    lastIndex = pointsCount - 1;
    return newNode;
}
```

FunctionNode addNodeByIndex(int index), добавляющий новый элемент в указанную позицию списка и возвращающий ссылку на объект этого элемента.

```
private FunctionNode addNodeByIndex(int index){  1 usage  new *
    FunctionNode node = (index == pointsCount) ? head : getNodeByIndex(index);
    FunctionNode newNode = new FunctionNode( point: null,node.previous,node);
    node.previous.next=newNode;
    node.previous=newNode;

    pointsCount++;
    lastIndex=index;
    lastNode=newNode;
    return newNode;
}
```

FunctionNode deleteNodeByIndex(int index)

```
private FunctionNode deleteNodeByIndex(int index){  1 usage  new *
    FunctionNode nodeDel = getNodeByIndex(index);
    nodeDel.previous.next= nodeDel.next;
    nodeDel.next.previous= nodeDel.previous;
    pointsCount--;
    if (lastIndex == index) {
        lastNode = (index == pointsCount) ? head.previous : nodeDel.next;
        lastIndex = (index == pointsCount) ? pointsCount - 1 : index;
    } else if (lastIndex > index) {
        lastIndex--;
    }
    return nodeDel;
}
```

Задание 5

Далее я реализовал методы и конструкторы в классе LinkedListTabulatedFunction, аналогичные методам и конструкторам класса TabulatedFunction.

```
    public LinkedListTabulatedFunction(double leftX,double rightX, int pointsCount){ 1 usage  new *
        if(leftX - rightX > - EPSILON){
            throw new IllegalArgumentException("Левая граница диапазона больше или равна правой");
        }
        else if (pointsCount<2) {
            throw new IllegalArgumentException("Количество предлагаемых точек меньше двух");
        }

        initializeList();
        double step = (rightX-leftX)/(pointsCount-1);
        for (int i =0; i<pointsCount; i++){
            addNodeToTail().point= new FunctionPoint( x: leftX+i*step, y: 0);
        }
    }

    public LinkedListTabulatedFunction(double leftX, double rightX, double[] values){ 1 usage  new *
        if(leftX - rightX > - EPSILON){
            throw new IllegalArgumentException("Левая граница диапазона больше или равна правой");
        }
        else if (values.length < 2) {
            throw new IllegalArgumentException("Количество предлагаемых точек меньше двух");
        }

        initializeList();
        double step = (rightX-leftX)/(values.length-1);
        for (int i =0; i<values.length; i++){
            addNodeToTail().point= new FunctionPoint( x: leftX+i*step,values[i]);
        }
    }

    public double getLeftDomainBorder(){ 1 usage  new *
        return head.next.point.getX();
    }

    public double getRightDomainBorder(){ 1 usage  new *
        return head.previous.point.getX();
    }
```

```
public double getFunctionValue(double x) {  2 usages  new *
    FunctionNode node = head.next;
    while (node != head) {
        if (Math.abs(x - node.point.getX()) < EPSILON) {
            return node.point.getY();
        }
        else if (node.next != head && node.point.getX() - x < -EPSILON && node.next.point.getX() - x > EPSILON) {
            double leftX = node.point.getX();
            double rightX = node.next.point.getX();
            double leftY = node.point.getY();
            double rightY = node.next.point.getY();
            return leftY + (rightY - leftY) * (x - leftX) / (rightX - leftX);
        }
        node = node.next;
    }
    return Double.NaN;
}

public int getPointsCount() { return pointsCount; }

public FunctionPoint getPoint(int index){  1 usage  new *
    if (index<0||index>=pointsCount){
        throw new FunctionPointIndexOutOfBoundsException("Введённый индекс выходит за границы набора точек или меньше нуля");
    }
    return getNodeByIndex(index).point;
}
```

```java
public void setPoint(int index, FunctionPoint point) throws InappropriateFunctionPointException {  3 usages  new *
    if (index<0||index>=pointsCount){
        throw new FunctionPointIndexOutOfBoundsException("Введённый индекс выходит за границы набора точек или меньше нуля");
    }

    else if (index == 0){
        if (point.getX() - getNodeByIndex(1).point.getX() > -EPSILON) {
            throw new InappropriateFunctionPointException("Координата x задаваемой точки лежит вне интервала");
        }
    }

    else if (index == pointsCount - 1){
        if (point.getX() - getNodeByIndex(pointsCount - 2).point.getX() < EPSILON) {
            throw new InappropriateFunctionPointException("Координата x задаваемой точки лежит вне интервала");
        }
    }

    else if (point.getX() - getNodeByIndex(index - 1).point.getX() < EPSILON || point.getX() - getNodeByIndex(index + 1).point.getX() > -EPSILON) {
        throw new InappropriateFunctionPointException("Координата x задаваемой точки лежит вне интервала");
    }

    getNodeByIndex(index).point = new FunctionPoint(point);
}

public double getPointX(int index){  18 usages  new *
    if (index<0||index>=pointsCount){
        throw new FunctionPointIndexOutOfBoundsException("Введённый индекс выходит за границы набора точек или меньше нуля");
    }
    return getNodeByIndex(index).point.getX();
}

public void setPointX(int index, double x) throws InappropriateFunctionPointException{  3 usages  new *
    if (index<0||index>=pointsCount){
        throw new FunctionPointIndexOutOfBoundsException("Введённый индекс выходит за границы набора точек или меньше нуля");
    }
```

```java
public void setPointX(int index, double x) throws InappropriateFunctionPointException{  3 usages  new *
    if (index<0||index>=pointsCount){
        throw new FunctionPointIndexOutOfBoundsException("Введённый индекс выходит за границы набора точек или меньше нуля");
    }

    else if (index == 0){
        if (x - getNodeByIndex(1).point.getX() > -EPSILON){
            throw new InappropriateFunctionPointException("Координата x задаваемой точки лежит вне интервала");
        }
    }

    else if (index == pointsCount - 1 ) {
        if(x - getNodeByIndex(pointsCount - 2).point.getX() < EPSILON) {
            throw new InappropriateFunctionPointException("Координата x задаваемой точки лежит вне интервала");
        }
    }

    else if (x - getNodeByIndex(index - 1).point.getX() < EPSILON || x - getNodeByIndex(index + 1).point.getX() > -EPSILON) {
        throw new InappropriateFunctionPointException("Координата x задаваемой точки лежит вне интервала");
    }

    getNodeByIndex(index).point.setX(x);
}

public double getPointY(int index){  16 usages  new *
    if (index<0||index>=pointsCount){
        throw new FunctionPointIndexOutOfBoundsException("Введённый индекс выходит за границы набора точек или меньше нуля");
    }
    return getNodeByIndex(index).point.getY();
}

public void setPointY(int index, double y){  3 usages  new *
    if (index<0||index>=pointsCount){
        throw new FunctionPointIndexOutOfBoundsException("Введённый индекс выходит за границы набора точек или меньше нуля");
    }
    getNodeByIndex(index).point.setY(y);
}
```

```java
public void deletePoint(int index){  2 usages  new *
    if(pointsCount<3){
        throw new IllegalStateException("В массиве находятся только две точки");
    }
    if (index<0||index>=pointsCount){
        throw new FunctionPointIndexOutOfBoundsException("Введённый индекс выходит за границы набора точек или меньше нуля");
    }
    deleteNodeByIndex(index-1);
}

public void addPoint(FunctionPoint point) throws InappropriateFunctionPointException{  3 usages  new *
    FunctionNode node = head.next;
    while (node != head) {
        if (Math.abs(node.point.getX() - point.getX()) < EPSILON) {
            throw new InappropriateFunctionPointException("Точка с такой координатой x уже существует");
        }
        node = node.next;
    }

    int insertIndex = 0;
    node = head.next;
    while (node != head && point.getX() - node.point.getX() > EPSILON) {
        node = node.next;
        insertIndex++;
    }
    FunctionNode newNode = addNodeByIndex(insertIndex);
    newNode.point = new FunctionPoint(point);
}
```

Задание 6

Далее я переименовал класс TabulatedFunction в ArrayTabulatedFunction и создал интерфейс TabulatedFunction для классов ArrayTabulatedFunction и LinkedListTabulatedFunction.

```java
package functions;

public interface TabulatedFunction {  3 usages  2 implementations  Юрий *
    public double getLeftDomainBorder();  1 usage  2 implementations  new *
    public double getRightDomainBorder();  1 usage  2 implementations  new *
    public double getFunctionValue(double x);  2 usages  2 implementations  new *
    public int getPointsCount();  21 usages  2 implementations  new *
    public FunctionPoint getPoint(int index);  1 usage  2 implementations  new *
    public void setPoint(int index, FunctionPoint point) throws InappropriateFunctionPointException;
    public double getPointX(int index);  18 usages  2 implementations  new *
    public void setPointX(int index, double x) throws InappropriateFunctionPointException;  3 usages  2
    public double getPointY(int index);  16 usages  2 implementations  new *
    public void setPointY(int index, double y);  3 usages  2 implementations  new *
    public void deletePoint(int index);  2 usages  2 implementations  new *
    public void addPoint(FunctionPoint point) throws InappropriateFunctionPointException;  3 usages  2 i
}
```

Задание 7

Переписал main для проверки всех методов

```java
import functions.*;

class lab_3 {  Юрий *
    static public void main(String[] args) throws InappropriateFunctionPointException {  new *
        ArrayTabulatedFunction ATF = new ArrayTabulatedFunction( leftX: 0, rightX: 10, pointsCount: 5);
        System.out.println("ПРОВЕРКА ArrayTabulatedFunction:");
        TestTabulatedFunction(ATF);

        LinkedListTabulatedFunction LLTF = new LinkedListTabulatedFunction( leftX: 0, rightX: 10, pointsCount: 5);
        double[] values = new double[5];
        for (int i = 0; i<5;i++){
            values[i]=Math.sqrt(LLTF.getPointX(i));
        }
        LinkedListTabulatedFunction LLTF1 = new LinkedListTabulatedFunction( leftX: 0, rightX: 10,values);
        System.out.println("ПРОВЕРКА LinkedListTabulatedFunction:");
        System.out.println("Проверка конструктора с values[]: ");
        for (int i = 0; i<5;i++){
            System.out.printf("f%d = (%.1f, %.3f)%n", i, LLTF1.getPointX(i),LLTF1.getPointY(i));
        }
        TestTabulatedFunction(LLTF);
    }
    static void TestTabulatedFunction(TabulatedFunction function) throws InappropriateFunctionPointException {  2 usages  Юрий *
        System.out.println("Функция root(x)");
        System.out.println("Область определения: [" + function.getLeftDomainBorder() + "; " + function.getRightDomainBorder() + "]");
        System.out.println("Количество точек: " + function.getPointsCount());

        System.out.println("\nТочки функции:");
        for (int i = 0; i < 5; i++) {
            function.setPointY(i, Math.sqrt(function.getPointX(i)));
            System.out.printf("f%d: (%.1f; %.3f)%n", i + 1, function.getPointX(i), function.getPointY(i));
        }

        System.out.println("\nЗначения f(x)  разных точках: ");
        double[] test = {1, 2.1, 11, -35, 632, 453.1, 5, 6, 7, 0};
        for (double x : test) {
            if (Double.isNaN(function.getFunctionValue(x))) {
                System.out.printf("f(%.1f): не определенно%n", x);
            } else {
                System.out.printf("f(%.1f) = %.3f%n", x, function.getFunctionValue(x));
            }
        }

        System.out.println("\nДобавление точки (5.55, sqrt(5.55)): ");
        FunctionPoint test_point = new FunctionPoint( x: 5.55, Math.sqrt(5.55));
        function.addPoint(test_point);
        System.out.println("Точки функции:");
        for (int i = 0; i < function.getPointsCount(); i++) {
            System.out.printf("f%d: (%.1f; %.3f)%n", i + 1, function.getPointX(i), function.getPointY(i));
        }

        System.out.println("\nДобавление точки (9, sqrt(9)):");
        test_point = new FunctionPoint( x: 9, Math.sqrt(9));
        function.addPoint(test_point);
        System.out.println("Точки функции:");
        for (int i = 0; i < function.getPointsCount(); i++) {
            System.out.printf("f%d: (%.1f; %.3f)%n", i + 1, function.getPointX(i), function.getPointY(i));
        }

        System.out.println("\nУдаление точки с номером 5:");
        System.out.printf("Точка f5 = (%.1f, %.3f)%n", function.getPointX( index: 4), function.getPointY( index: 4));
        function.deletePoint( index: 5);
        System.out.println("Точки функции:");
        for (int i = 0; i < function.getPointsCount(); i++) {
            System.out.printf("f%d: (%.1f; %.3f)%n", i + 1, function.getPointX(i), function.getPointY(i));
        }

        System.out.println("\nЗамена точки с номером 3 на точку f = (4,sqrt(4))");
        test_point = new FunctionPoint( x: 4, Math.sqrt(4));
        System.out.printf("Исходная точка: f3'= (%.1f; %.3f)%n", function.getPointX( index: 2), function.getPointY( index: 2));
        function.setPoint( index: 2, test_point);
        System.out.printf("Измененная точка: f3'= (%.1f; %.3f)%n", function.getPointX( index: 2), function.getPointY( index: 2));
        System.out.println("Точки функции:");
        for (int i = 0; i < function.getPointsCount(); i++) {
            System.out.printf("f%d: (%.1f; %.3f)%n", i + 1, function.getPointX(i), function.getPointY(i));
        }

        System.out.println("\nЗамена точки с номером 6 по значению x = 9,5");
        System.out.printf("Исходная точка: f6'= (%.1f; %.3f)%n", function.getPointX( index: 5), function.getPointY( index: 5));
```

```
        function.setPointX( index: 5,  x: 9.5);
        function.setPointY( index: 5, Math.sqrt(9.5));
        System.out.printf("Измененная точка: f6'= (%.1f; %.3f)%n", function.getPointX( index: 5), function.getPointY( index: 5));
        System.out.println("Точки функции:");
        for (int i = 0; i < function.getPointsCount(); i++) {
            System.out.printf("f%d: (%.1f; %.3f)%n", i + 1, function.getPointX(i), function.getPointY(i));
        }

        System.out.println("\nПроверка исключений:");
        try{
            System.out.println("   Проверка метода getPoint:\n" +
                    "       Попытка получить элемент с индексом "+function.getPointsCount());
            function.getPoint(function.getPointsCount());
            System.out.println("Метод успешно сработал");
        }
        catch (FunctionPointIndexOutOfBoundsException e){
            System.out.println("       Метод выдал ошибку: "+ e.getMessage());
        }

        try{
            System.out.println("\n   Проверка метода setPoint:\n" +
                    "       Попытка внести элемент с индексом "+function.getPointsCount());
            function.setPoint(function.getPointsCount(), new FunctionPoint( x: 0, y: 0));
            System.out.println("Метод успешно сработал");
        }
        catch (FunctionPointIndexOutOfBoundsException e){
            System.out.println("       Метод выдал ошибку: "+ e.getMessage());
        }

        try{
            System.out.println("\n       Попытка внести точку F = (1, 1)");
            function.setPoint( index: 3, new FunctionPoint( x: 1, y: 1));
            System.out.println("    Метод успешно сработал");
        }
        catch (InappropriateFunctionPointException e){
            System.out.println("       Метод выдал ошибку: "+ e.getMessage());
        }
```

```
        try{
            System.out.println("\n   Проверка метода getPointX:\n" +
                    "       Попытка вывести значение x точки с индексом "+function.getPointsCount());
            function.getPointX(function.getPointsCount());
            System.out.println("Метод успешно сработал");
        }
        catch (FunctionPointIndexOutOfBoundsException e){
            System.out.println("       Метод выдал ошибку: "+ e.getMessage());
        }

        try{
            System.out.println("\n   Проверка метода setPointX:\n" +
                    "       Попытка внести значение x элемента с индексом "+function.getPointsCount());
            function.setPointX(function.getPointsCount(),  x: 0);
            System.out.println("Метод успешно сработал");
        }
        catch (FunctionPointIndexOutOfBoundsException e){
            System.out.println("       Метод выдал ошибку: "+ e.getMessage());
        }

        try{
            System.out.println("\n       Попытка внести значение x = 1000 ");
            function.setPointX( index: 3, x: 1000);
            System.out.println("    Метод успешно сработал");
        }
        catch (InappropriateFunctionPointException e){
            System.out.println("       Метод выдал ошибку: "+ e.getMessage());
        }

        try{
            System.out.println("\n   Проверка метода getPointY:\n" +
                    "       Попытка вывести значение y точки с индексом "+function.getPointsCount());
            function.getPointY(function.getPointsCount());
            System.out.println("Метод успешно сработал");
        }
        catch (FunctionPointIndexOutOfBoundsException e){
            System.out.println("       Метод выдал ошибку: "+ e.getMessage());
        }
```

```
        try{
            System.out.println("\n   Проверка метода setPointY:\n" +
                    "       Попытка внести значение Y элемента с индексом "+function.getPointsCount());
            function.setPointY(function.getPointsCount(), y: 0);
            System.out.println("Метод успешно сработал");
        }
        catch (FunctionPointIndexOutOfBoundsException e){
            System.out.println("       Метод выдал ошибку: "+ e.getMessage());
        }

        try{
            System.out.println("\n   Проверка метода deletePoint:\n" +
                    "       Попытка удалить точку с индексом "+function.getPointsCount());
            function.deletePoint(function.getPointsCount());
            System.out.println("Метод успешно сработал");
        }
        catch (FunctionPointIndexOutOfBoundsException e){
            System.out.println("       Метод выдал ошибку: "+ e.getMessage());
        }

        try{
            System.out.println("\n   Проверка метода addPoint:\n");
            System.out.printf("       Попытка добавить точку F (%.1f, %.3f):%n", function.getPointX( index: 1),function.getPointY( index: 1));
            function.addPoint(new FunctionPoint(function.getPointX( index: 1),function.getPointY( index: 1)));
            System.out.println("Метод успешно сработал");
        }
        catch (InappropriateFunctionPointException e){
            System.out.println("       Метод выдал ошибку: "+ e.getMessage());
        }
    }
}
```

ПРОВЕРКА ArrayTF

```
Функция root(x)
Область определения: [0.0; 10.0]
Количество точек: 5

Точки функции:
f1: (0,0; 0,000)
f2: (2,5; 1,581)
f3: (5,0; 2,236)
f4: (7,5; 2,739)
f5: (10,0; 3,162)

Значения f(x)  разных точках:
f(1,0) = 0,632
f(2,1) = 1,328
f(11,0): не определенно
f(-35,0): не определенно
f(632,0): не определенно
f(453,1): не определенно
f(5,0) = 2,236
f(6,0) = 2,437
f(7,0) = 2,638
f(0,0) = 0,000

Добавление точки (5.55, sqrt(5.55)):
Точки функции:
f1: (0,0; 0,000)
f2: (2,5; 1,581)
f3: (5,0; 2,236)
f4: (5,6; 2,356)
f5: (7,5; 2,739)
f6: (10,0; 3,162)

Добавление точки (9, sqrt(9)):
Точки функции:
f1: (0,0; 0,000)
f2: (2,5; 1,581)
f3: (5,0; 2,236)
```

ПРОВЕРКА LinkedListTF

```
Функция root(x)
Область определения: [0.0; 10.0]
Количество точек: 5

Точки функции:
f1: (0,0; 0,000)
f2: (2,5; 1,581)
f3: (5,0; 2,236)
f4: (7,5; 2,739)
f5: (10,0; 3,162)

Значения f(x)  разных точках:
f(1,0) = 0,632
f(2,1) = 1,328
f(11,0): не определенно
f(-35,0): не определенно
f(632,0): не определенно
f(453,1): не определенно
f(5,0) = 2,236
f(6,0) = 2,437
f(7,0) = 2,638
f(0,0) = 0,000

Добавление точки (5.55, sqrt(5.55)):
Точки функции:
f1: (0,0; 0,000)
f2: (2,5; 1,581)
f3: (5,0; 2,236)
f4: (5,6; 2,356)
f5: (7,5; 2,739)
f6: (10,0; 3,162)

Добавление точки (9, sqrt(9)):
Точки функции:
f1: (0,0; 0,000)
f2: (2,5; 1,581)
f3: (5,0; 2,236)
f4: (5,6; 2,356)
f5: (7,5; 2,739)
```

```
f4: (5,6; 2,356)
f5: (7,5; 2,739)
f6: (9,0; 3,000)
f7: (10,0; 3,162)

Удаление точки с номером 5:
Точка f5 = (7,5, 2,739)
Точки функции:
f1: (0,0; 0,000)
f2: (2,5; 1,581)
f3: (5,0; 2,236)
f4: (5,6; 2,356)
f5: (9,0; 3,000)
f6: (10,0; 3,162)

Замена точки с номером 3 на точку f = (4,sqrt(4))
Исходная точка: f3'= (5,0; 2,236)
Измененная точка: f3'= (4,0; 2,000)
Точки функции:
f1: (0,0; 0,000)
f2: (2,5; 1,581)
f3: (4,0; 2,000)
f4: (5,6; 2,356)
f5: (9,0; 3,000)
f6: (10,0; 3,162)

Замена точки с номером 6 по значению x = 9,5
Исходная точка: f6'= (10,0; 3,162)
Измененная точка: f6'= (9,5; 3,082)
Точки функции:
f1: (0,0; 0,000)
f2: (2,5; 1,581)
f3: (4,0; 2,000)
f4: (5,6; 2,356)
f5: (9,0; 3,000)
f6: (9,5; 3,082)
```

```
f6: (9,0; 3,000)
f7: (10,0; 3,162)

Удаление точки с номером 5:
Точка f5 = (7,5, 2,739)
Точки функции:
f1: (0,0; 0,000)
f2: (2,5; 1,581)
f3: (5,0; 2,236)
f4: (5,6; 2,356)
f5: (9,0; 3,000)
f6: (10,0; 3,162)

Замена точки с номером 3 на точку f = (4,sqrt(4))
Исходная точка: f3'= (5,0; 2,236)
Измененная точка: f3'= (4,0; 2,000)
Точки функции:
f1: (0,0; 0,000)
f2: (2,5; 1,581)
f3: (4,0; 2,000)
f4: (5,6; 2,356)
f5: (9,0; 3,000)
f6: (10,0; 3,162)

Замена точки с номером 6 по значению x = 9,5
Исходная точка: f6'= (10,0; 3,162)
Измененная точка: f6'= (9,5; 3,082)
Точки функции:
f1: (0,0; 0,000)
f2: (2,5; 1,581)
f3: (4,0; 2,000)
f4: (5,6; 2,356)
f5: (9,0; 3,000)
f6: (9,5; 3,082)
```

Один и тот же метод для объектов разных классов с одинаковыми значениями выдал одинаковый результат, что означает правильность написания LinkedListTabulatedFunction.

Проверка исключений ArrayTF

```
Проверка исключений:
   Проверка метода getPoint:
       Попытка получить элемент с индексом 6
       Метод выдал ошибку: Введённый индекс выходит за границы набора точек или меньше нуля

   Проверка метода setPoint:
       Попытка внести элемент с индексом 6
       Метод выдал ошибку: Введённый индекс выходит за границы набора точек или меньше нуля

       Попытка внести точку F = (1, 1)
       Метод выдал ошибку: Координата x задаваемой точки лежит вне интервала

   Проверка метода getPointX:
       Попытка вывести значение x точки с индексом 6
       Метод выдал ошибку: Введённый индекс выходит за границы набора точек или меньше нуля

   Проверка метода setPointX:
       Попытка внести значение x элемента с индексом 6
       Метод выдал ошибку: Введённый индекс выходит за границы набора точек или меньше нуля

       Попытка внести значение x = 1000
    Метод успешно сработал

   Проверка метода getPointY:
       Попытка вывести значение y точки с индексом 6
       Метод выдал ошибку: Введённый индекс выходит за границы набора точек или меньше нуля

   Проверка метода setPointY:
       Попытка внести значение Y элемента с индексом 6
       Метод выдал ошибку: Введённый индекс выходит за границы набора точек или меньше нуля

   Проверка метода deletePoint:
       Попытка удалить точку с индексом 6
       Метод выдал ошибку: Введённый индекс выходит за границы набора точек или меньше нуля

   Проверка метода addPoint:

       Попытка добавить точку F (2,5, 1,581):
       Метод выдал ошибку: В наборе точек функции есть точка, абсцисса которой совпадает с абсциссой добавляемой точки
```

Проверка исключений LinkedListTF

```
Проверка исключений:
   Проверка метода getPoint:
       Попытка получить элемент с индексом 6
       Метод выдал ошибку: Введённый индекс выходит за границы набора точек или меньше нуля

   Проверка метода setPoint:
       Попытка внести элемент с индексом 6
       Метод выдал ошибку: Введённый индекс выходит за границы набора точек или меньше нуля

       Попытка внести точку F = (1, 1)
       Метод выдал ошибку: Координата x задаваемой точки лежит вне интервала

   Проверка метода getPointX:
       Попытка вывести значение x точки с индексом 6
       Метод выдал ошибку: Введённый индекс выходит за границы набора точек или меньше нуля

   Проверка метода setPointX:
       Попытка внести значение x элемента с индексом 6
       Метод выдал ошибку: Введённый индекс выходит за границы набора точек или меньше нуля

       Попытка внести значение x = 1000
       Метод выдал ошибку: Координата x задаваемой точки лежит вне интервала

   Проверка метода getPointY:
       Попытка вывести значение y точки с индексом 6
       Метод выдал ошибку: Введённый индекс выходит за границы набора точек или меньше нуля

   Проверка метода setPointY:
       Попытка внести значение Y элемента с индексом 6
       Метод выдал ошибку: Введённый индекс выходит за границы набора точек или меньше нуля

   Проверка метода deletePoint:
       Попытка удалить точку с индексом 6
       Метод выдал ошибку: Введённый индекс выходит за границы набора точек или меньше нуля

   Проверка метода addPoint:

       Попытка добавить точку F (2,5, 1,581):
       Метод выдал ошибку: Точка с такой координатой x уже существует
```

Проверка создания LinkedListTF через второй конструктор (те же значения)

```
Проверка конструктора с values[]:
f0 = (0,0, 0,000)
f1 = (2,5, 1,581)
f2 = (5,0, 2,236)
f3 = (7,5, 2,739)
f4 = (10,0, 3,162)
```