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»

Тема «Базовые конструкции»

Выполнил: Пантелеев Ю.В.

Группа: 6201-120303

Самара, 2025

Задание на лабораторную работу

Задание 1

Я создал пакет functions, в котором далее будут создаваться классы программы.

Задание 2

В пакете functions создал класс FunctionPoint и описал следующие конструкции:

```
package functions;

public class FunctionPoint {  16 usages  new *
    private double x;  5 usages
    private double y;  5 usages

    public FunctionPoint(double x1, double y1){  5 usages  new *
        x=x1;
        y=y1;
    }

    public double getX() {  16 usages  new *
        return x;
    }
    public double getY() {  4 usages  new *
        return y;
    }

    public void setX(double value) {  1 usage  new *
        x = value;
    }
    public void setY(double value){  1 usage  new *
        y=value;
    }

    public FunctionPoint(FunctionPoint point){  no usages  new *
        x= point.getX();
        y= point.getY();
    }

    public FunctionPoint(){  no usages  new *
        x=0;
        y=0;
    }
}
```

Задание 3

В пакете functions создал класс TabulatedFunction, объект которого должен описывать табулированную функцию. Для этого я прописал следующие методы:

```
package functions;

public class TabulatedFunction {  4 usages  new *
    private FunctionPoint[] points;  28 usages
    private int pointsCount;  17 usages

    public TabulatedFunction(double leftX,double rightX, int pointsCount){  1 usage  new *
        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( x1: leftX+i*step, y1: 0);
        }
    }

    public TabulatedFunction(double leftX, double rightX, double[] values){  1 usage  new *
        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( x1: leftX+i*step,values[i]);
        }
    }
```

Задание 4

В классе TabulatedFunction я описал методы по заданию, необходимые для работы с функцией.

```
public double getLeftDomainBorder(){  1 usage  Юрий
    return points[0].getX();
}
public double getRightDomainBorder(){  1 usage  Юрий
    return points[pointsCount-1].getX();
}
public double getFunctionValue(double x){  2 usages  Юрий *
    double LX ,RX; //LX-leftX, RX-rightX
    double LY ,RY; //LY-leftY, RY-rightY
    for (int i = 1; i<pointsCount-1;i++){
        LX= points[i-1].getX();
        RX= points[i+1].getX(); //LX-leftX, RX-rightX
        LY=points[i-1].getY();
        RY= points[pointsCount-1].getY(); //LY-leftY, RY-rightY
    if(LX<x && RX>x){
        return ((x-LX)*(RY-LY))/(RX-LX)+LY;
    }
    }
    return Double.NaN;
}
```

Задание 5

В классе TabulatedFunction я описал методы, необходимые для работы с точками табулированной функции.

```
public int getPointsCount(){  7 usages  Юрий
    return pointsCount;
}

public FunctionPoint getPoint(int index){  no usages  Юрий *
    FunctionPoint point = new FunctionPoint(points[index].getX(),points[index].getY());
    return point;
}

public void setPoint(int index, FunctionPoint point){  1 usage  Юрий *
    if (points[index-1].getX()<point.getX()&&points[index + 1].getX()>point.getX()){
        points[index]= new FunctionPoint(point.getX(),point.getY());
    }
}

public double getPointX(int index){  14 usages  Юрий
    return points[index].getX();
}

public void setPointX(int index, double x){  1 usage  Юрий
    if (points[index-1].getX()<x&&x<points[index+1].getX())
        points[index].setX(x);
}

public double getPointY(int index){  12 usages  Юрий
    return points[index].getY();
}

public void setPointY(int index, double y){  2 usages  Юрий
    points[index].setY(y);
}
```

S

Задание 6

В классе TabulatedFunction я описал методы, изменяющие количество точек табулированной функции.

```
    public void deletePoint(int index){  1 usage  new *
        System.arraycopy(points,index,points, destPos: index-1, length: pointsCount-index);
        points[pointsCount-1]=null;
        pointsCount--;
    }

    public void addPoint(FunctionPoint point) {  2 usages  new *
        for (int i = 1; i < pointsCount; i++) {
            if (point.getX() < points[i].getX() && point.getX() > points[i - 1].getX()) {
                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;
            }
        }
    }
}
```

Задание 7

Я проверил работу написанных классов.

1)Сперва я проверил первый конструктор в классе TabulatedFunction, сохраняя значения Y для проверки второго конструктора.

```
import functions.*;

class lab_2{  Юрий *
    static public void main(String[] args){  Юрий *
        TabulatedFunction f1 = new TabulatedFunction( leftX: 0, rightX: 10, pointsCount: 15);
        double[] y_value = new double[15];
        System.out.println("Функция root(x)");
        System.out.println("Область определения: [" + f1.getLeftDomainBorder() + "; " + f1.getRightDomainBorder() + "]");
        System.out.println("Количество точек: " + f1.getPointsCount());

        System.out.println("\nТочки функции:");
        for (int i=0;i<15;i++){
            f1.setPointY(i,Math.sqrt(f1.getPointX(i)));
            System.out.printf("f%d: (%.1f; %.3f)%n", i + 1, f1.getPointX(i), f1.getPointY(i));
            y_value[i]=Math.sqrt(f1.getPointX(i));
        }
```

Вывод 1:

```
"C:\Program Files\Java\jdk-24\bi
Функция 1/x
Область определения: [0.0; 10.0]
Количество точек: 15

Точки функции:
f1: (0,0; 0,000)
f2: (0,7; 0,845)
f3: (1,4; 1,195)
f4: (2,1; 1,464)
f5: (2,9; 1,690)
f6: (3,6; 1,890)
f7: (4,3; 2,070)
f8: (5,0; 2,236)
f9: (5,7; 2,390)
f10: (6,4; 2,535)
f11: (7,1; 2,673)
f12: (7,9; 2,803)
f13: (8,6; 2,928)
f14: (9,3; 3,047)
f15: (10,0; 3,162)
```

2)Проверка второго конструктора:

```
System.out.println("\nПроверка создания экземляра класса через второй конструктор: ");
TabulatedFunction f = new TabulatedFunction( leftX: 0, rightX: 10,y_value);
System.out.println("Точки функции:");
for (int i=0;i<f.getPointsCount();i++){
    System.out.printf("f'%d: (%.1f; %.3f)%n", i + 1, f.getPointX(i), f.getPointY(i));
}
```

Вывод 2:

```
Проверка создания экземляра класса через второй конструктор:
Точки функции:
f'1: (0,0; 0,000)
f'2: (0,7; 0,845)
f'3: (1,4; 1,195)
f'4: (2,1; 1,464)
f'5: (2,9; 1,690)
f'6: (3,6; 1,890)
f'7: (4,3; 2,070)
f'8: (5,0; 2,236)
f'9: (5,7; 2,390)
f'10: (6,4; 2,535)
f'11: (7,1; 2,673)
f'12: (7,9; 2,803)
f'13: (8,6; 2,928)
f'14: (9,3; 3,047)
f'15: (10,0; 3,162)
```

3)Проверка метода getFunctionValue для разных точек.

```
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(f.getFunctionValue(x))){
        System.out.printf("f'(%.1f): не определенно%n",x);
    }
    else{
        System.out.printf("f'(%.1f) = %.3f%n",x,f.getFunctionValue(x));
    }
}
```

Вывод 3:

```
Значения f'(x)  разных точках:
f'(1,0) = 0,316
f'(2,1) = 0,664
f'(11,0): не определенно
f'(-35,0): не определенно
f'(632,0): не определенно
f'(453,1): не определенно
f'(5,0) = 1,581
f'(6,0) = 1,897
f'(7,0) = 2,214
f'(0,0) = 0,000
```

4)Проверка метода addPoint на двух точках.

```
System.out.println("\nДобавление точки (5.55, sqrt(5.55)): ");
FunctionPoint test_point = new FunctionPoint( x1: 5.55,Math.sqrt(5.55));
f.addPoint(test_point);
System.out.println("Точки функции:");
for (int i=0;i<f.getPointsCount();i++){
    System.out.printf("f'%d: (%.1f; %.3f)%n", i + 1, f.getPointX(i), f.getPointY(i));
}

System.out.println("\nДобавление точки (9, sqrt(9)):");
test_point = new FunctionPoint( x1: 9,Math.sqrt(9));
f.addPoint(test_point);
System.out.println("Точки функции:");
for (int i=0;i<f.getPointsCount();i++){
    System.out.printf("f'%d: (%.1f; %.3f)%n", i + 1, f.getPointX(i), f.getPointY(i));
}
```

Вывод 4:

```
Добавление точки (5.55, sqrt(5.55)):
Точки функции:
f'1: (0,0; 0,000)
f'2: (0,7; 0,845)
f'3: (1,4; 1,195)
f'4: (2,1; 1,464)
f'5: (2,9; 1,690)
f'6: (3,6; 1,890)
f'7: (4,3; 2,070)
f'8: (5,0; 2,236)
f'9: (5,6; 2,356)
f'10: (5,7; 2,390)
f'11: (6,4; 2,535)
f'12: (7,1; 2,673)
f'13: (7,9; 2,803)
f'14: (8,6; 2,928)
f'15: (9,3; 3,047)
f'16: (10,0; 3,162)

Добавление точки (9, sqrt(9)):
Точки функции:
f'1: (0,0; 0,000)
f'2: (0,7; 0,845)
f'3: (1,4; 1,195)
f'4: (2,1; 1,464)
f'5: (2,9; 1,690)
f'6: (3,6; 1,890)
f'7: (4,3; 2,070)
f'8: (5,0; 2,236)
f'9: (5,6; 2,356)
f'10: (5,7; 2,390)
f'11: (6,4; 2,535)
f'12: (7,1; 2,673)
f'13: (7,9; 2,803)
f'14: (8,6; 2,928)
f'15: (9,0; 3,000)
f'16: (9,3; 3,047)
f'17: (10,0; 3,162)
```

5)Проверка метода detelePoint

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

Вывод 5:

```
Удаление точки с номером 5:
Точка f5 = (2,9, 1,690)
Точки функции:
f'1: (0,0; 0,000)
f'2: (0,7; 0,845)
f'3: (1,4; 1,195)
f'4: (2,1; 1,464)
f'5: (3,6; 1,890)
f'6: (4,3; 2,070)
f'7: (5,0; 2,236)
f'8: (5,6; 2,356)
f'9: (5,7; 2,390)
f'10: (6,4; 2,535)
f'11: (7,1; 2,673)
f'12: (7,9; 2,803)
f'13: (8,6; 2,928)
f'14: (9,0; 3,000)
f'15: (9,3; 3,047)
f'16: (10,0; 3,162)
```

6)Проверка метода setPoint

```
System.out.println("\nЗамена точки с номером 6 на точку f = (4,sqrt(4))");
test_point= new FunctionPoint( x1: 4,Math.sqrt(4));
System.out.printf("Исходная точка: f6'= (%.1f; %.3f)%n", f.getPointX( index: 5), f.getPointY( index: 5));
f.setPoint( index: 5,test_point);
System.out.printf("Измененная точка: f6'= (%.1f; %.3f)%n", f.getPointX( index: 5), f.getPointY( index: 5));
System.out.println("Точки функции:");
for (int i=0;i<f.getPointsCount();i++){
    System.out.printf("f'%d: (%.1f; %.3f)%n", i + 1, f.getPointX(i), f.getPointY(i));
}
```

Вывод 6:

```
Замена точки с номером 6 на точку f = (4,sqrt(4))
Исходная точка: f6'= (4,3; 2,070)
Измененная точка: f6'= (4,0; 2,000)
Точки функции:
f'1: (0,0; 0,000)
f'2: (0,7; 0,845)
f'3: (1,4; 1,195)
f'4: (2,1; 1,464)
f'5: (3,6; 1,890)
f'6: (4,0; 2,000)
f'7: (5,0; 2,236)
f'8: (5,6; 2,356)
f'9: (5,7; 2,390)
f'10: (6,4; 2,535)
f'11: (7,1; 2,673)
f'12: (7,9; 2,803)
f'13: (8,6; 2,928)
f'14: (9,0; 3,000)
f'15: (9,3; 3,047)
f'16: (10,0; 3,162)
```

7)Проверка метода setPointX

```
        System.out.println("\nЗамена точки с номером 10 по значению x = 6");
        System.out.printf("Исходная точка: f10'= (%.1f; %.3f)%n", f.getPointX( index: 9), f.getPointY( index: 9));
        f.setPointX( index: 9, x: 6);
        f.setPointY( index: 9,Math.sqrt(6));
        System.out.printf("Измененная точка: f10'= (%.1f; %.3f)%n", f.getPointX( index: 9), f.getPointY( index: 9));
        System.out.println("Точки функции:");
        for (int i=0;i<f.getPointsCount();i++){
            System.out.printf("f'%d: (%.1f; %.3f)%n", i + 1, f.getPointX(i), f.getPointY(i));
        }
    }
}
```

Вывод 7:

```
Замена точки с номером 10 по значению x = 6
Исходная точка: f10'= (6,4; 2,535)
Измененная точка: f10'= (6,0; 2,449)
Точки функции:
f'1: (0,0; 0,000)
f'2: (0,7; 0,845)
f'3: (1,4; 1,195)
f'4: (2,1; 1,464)
f'5: (3,6; 1,890)
f'6: (4,0; 2,000)
f'7: (5,0; 2,236)
f'8: (5,6; 2,356)
f'9: (5,7; 2,390)
f'10: (6,0; 2,449)
f'11: (7,1; 2,673)
f'12: (7,9; 2,803)
f'13: (8,6; 2,928)
f'14: (9,0; 3,000)
f'15: (9,3; 3,047)
f'16: (10,0; 3,162)
```

Общие выводы

Класс FunctionPoint:

```java
package functions;

public class FunctionPoint {  16 usages  new *
    private double x;  5 usages
    private double y;  5 usages

    public FunctionPoint(double x1, double y1){  5 usages  new *
        x=x1;
        y=y1;
    }

    public double getX() {  16 usages  new *
        return x;
    }
    public double getY() {  4 usages  new *
        return y;
    }

    public void setX(double value) {  1 usage  new *
        x = value;
    }
    public void setY(double value){  1 usage  new *
        y=value;
    }

    public FunctionPoint(FunctionPoint point){  no usages  new *
        x= point.getX();
        y= point.getY();
    }

    public FunctionPoint(){  no usages  new *
        x=0;
        y=0;
    }
}

```

Класс TabulatedFunction:

```java
package functions;

public class TabulatedFunction {  4 usages  Юрий *
    private FunctionPoint[] points;  29 usages
    private int pointsCount;  17 usages

    public TabulatedFunction(double leftX,double rightX, int pointsCount){  17 usages  Юрий
        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( x1: leftX+i*step, y1: 0);
        }
    }

    public TabulatedFunction(double leftX, double rightX, double[] values){  17 usages  Юрий
        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( x1: leftX+i*step,values[i]);
        }
    }

    public double getLeftDomainBorder(){  1 usage  Юрий
        return points[0].getX();
    }
    public double getRightDomainBorder(){  1 usage  Юрий
        return points[pointsCount-1].getX();
    }
    public double getFunctionValue(double x){  2 usages  Юрий *
        double LX ,RX; //LX-leftX, RX-rightX
        double LY ,RY; //LY-leftY, RY-rightY
        for (int i = 1; i<pointsCount-1;i++){
            LX= points[i-1].getX();
            RX= points[i+1].getX(); //LX-leftX, RX-rightX
            LY=points[i-1].getY();
```

```
            RY= points[i+1].getY(); //LY-leftY, RY-rightY
        if(LX<x && RX>x){
            return ((x-LX)*(RY-LY))/(RX-LX)+LY;
        }
        }
        return Double.NaN;
    }

    public int getPointsCount(){  7 usages  Юрий
        return pointsCount;
    }

    public FunctionPoint getPoint(int index){  no usages  Юрий *
        FunctionPoint point = new FunctionPoint(points[index].getX(),points[index].getY());
        return point;
    }

@   public void setPoint(int index, FunctionPoint point){  1 usage  Юрий *
        if (points[index-1].getX()<point.getX()&&points[index + 1].getX()>point.getX()){
            points[index]= new FunctionPoint(point.getX(),point.getY());
        }
    }

    public double getPointX(int index){  14 usages  Юрий
        return points[index].getX();
    }

    public void setPointX(int index, double x){  1 usage  Юрий
        if (points[index-1].getX()<x&&x<points[index+1].getX())
            points[index].setX(x);
    }

    public double getPointY(int index){  12 usages  Юрий
        return points[index].getY();
    }
    public void setPointY(int index, double y){  2 usages  Юрий
        points[index].setY(y);
    }

    public void deletePoint(int index){  1 usage  Юрий
        System.arraycopy(points,index,points, destPos: index-1, length: pointsCount-index);
        points[pointsCount-1]=null;
        pointsCount--;
    }

    public void addPoint(FunctionPoint point) {  2 usages  Юрий
        for (int i = 1; i < pointsCount; i++) {
            if (point.getX() < points[i].getX() && point.getX() > points[i - 1].getX()) {
                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;
            }
        }
    }
}
```

Класс lab_2:

```java
import functions.*;

class lab_2{  Юрий *
    static public void main(String[] args){  Юрий *
        TabulatedFunction f1 = new TabulatedFunction( leftX: 0, rightX: 10, pointsCount: 15);
        double[] y_value = new double[15];
        System.out.println("Функция root(x)");
        System.out.println("Область определения: [" + f1.getLeftDomainBorder() + "; " + f1.getRightDomainBorder() + "]");
        System.out.println("Количество точек: " + f1.getPointsCount());

        System.out.println("\nТочки функции:");
        for (int i=0;i<15;i++){
            f1.setPointY(i,Math.sqrt(f1.getPointX(i)));
            System.out.printf("f%d: (%.1f; %.3f)%n", i + 1, f1.getPointX(i), f1.getPointY(i));
            y_value[i]=Math.sqrt(f1.getPointX(i));
        }

        System.out.println("\nПроверка создания экземляра класса через второй конструктор: ");
        TabulatedFunction f = new TabulatedFunction( leftX: 0, rightX: 10,y_value);
        System.out.println("Точки функции:");
        for (int i=0;i<f.getPointsCount();i++){
            System.out.printf("f'%d: (%.1f; %.3f)%n", i + 1, f.getPointX(i), f.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(f.getFunctionValue(x))){
                System.out.printf("f'(%.1f): не определенно%n",x);
            }
            else{
                System.out.printf("f'(%.1f) = %.3f%n",x,f.getFunctionValue(x));
            }
        }

        System.out.println("\nДобавление точки (5.55, sqrt(5.55)): ");
        FunctionPoint test_point = new FunctionPoint( x1: 5.55,Math.sqrt(5.55));
        f.addPoint(test_point);
        System.out.println("Точки функции:");
        for (int i=0;i<f.getPointsCount();i++){
```

```
            System.out.printf("f'%d: (%.1f; %.3f)%n", i + 1, f.getPointX(i), f.getPointY(i));
        }

        System.out.println("\nДобавление точки (9, sqrt(9)):");
        test_point = new FunctionPoint( x1: 9,Math.sqrt(9));
        f.addPoint(test_point);
        System.out.println("Точки функции:");
        for (int i=0;i<f.getPointsCount();i++){
            System.out.printf("f'%d: (%.1f; %.3f)%n", i + 1, f.getPointX(i), f.getPointY(i));
        }

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

        System.out.println("\nЗамена точки с номером 6 на точку f = (4,sqrt(4))");
        test_point= new FunctionPoint( x1: 4,Math.sqrt(4));
        System.out.printf("Исходная точка: f6'= (%.1f; %.3f)%n", f.getPointX( index: 5), f.getPointY( index: 5));
        f.setPoint( index: 5,test_point);
        System.out.printf("Измененная точка: f6'= (%.1f; %.3f)%n", f.getPointX( index: 5), f.getPointY( index: 5));
        System.out.println("Точки функции:");
        for (int i=0;i<f.getPointsCount();i++){
            System.out.printf("f'%d: (%.1f; %.3f)%n", i + 1, f.getPointX(i), f.getPointY(i));
        }

        System.out.println("\nЗамена точки с номером 10 по значению x = 6");
        System.out.printf("Исходная точка: f10'= (%.1f; %.3f)%n", f.getPointX( index: 9), f.getPointY( index: 9));
        f.setPointX( index: 9, x: 6);
        f.setPointY( index: 9,Math.sqrt(6));
        System.out.printf("Измененная точка: f10'= (%.1f; %.3f)%n", f.getPointX( index: 9), f.getPointY( index: 9));
        System.out.println("Точки функции:");
        for (int i=0;i<f.getPointsCount();i++){
            System.out.printf("f'%d: (%.1f; %.3f)%n", i + 1, f.getPointX(i), f.getPointY(i));
        }
```

Вывод в консоли:

```
"C:\Program Files\Java\jdk-24\bin\java.exe" "-javaagent:C:\P
Функция root(x)
Область определения: [0.0; 10.0]
Количество точек: 15

Точки функции:
f1: (0,0; 0,000)
f2: (0,7; 0,845)
f3: (1,4; 1,195)
f4: (2,1; 1,464)
f5: (2,9; 1,690)
f6: (3,6; 1,890)
f7: (4,3; 2,070)
f8: (5,0; 2,236)
f9: (5,7; 2,390)
f10: (6,4; 2,535)
f11: (7,1; 2,673)
f12: (7,9; 2,803)
f13: (8,6; 2,928)
f14: (9,3; 3,047)
f15: (10,0; 3,162)

Проверка создания экземляра класса через второй конструктор:
Точки функции:
f'1: (0,0; 0,000)
f'2: (0,7; 0,845)
f'3: (1,4; 1,195)
f'4: (2,1; 1,464)
f'5: (2,9; 1,690)
f'6: (3,6; 1,890)
f'7: (4,3; 2,070)
f'8: (5,0; 2,236)
f'9: (5,7; 2,390)
f'10: (6,4; 2,535)
f'11: (7,1; 2,673)
f'12: (7,9; 2,803)
f'13: (8,6; 2,928)
f'14: (9,3; 3,047)
f'15: (10,0; 3,162)
```

```
Значения f'(x)  разных точках:
f'(1,0) = 0,837
f'(2,1) = 1,445
f'(11,0): не определенно
f'(-35,0): не определенно
f'(632,0): не определенно
f'(453,1): не определенно
f'(5,0) = 2,230
f'(6,0) = 2,446
f'(7,0) = 2,644
f'(0,0): не определенно

Добавление точки (5.55, sqrt(5.55)):
Точки функции:
f'1: (0,0; 0,000)
f'2: (0,7; 0,845)
f'3: (1,4; 1,195)
f'4: (2,1; 1,464)
f'5: (2,9; 1,690)
f'6: (3,6; 1,890)
f'7: (4,3; 2,070)
f'8: (5,0; 2,236)
f'9: (5,6; 2,356)
f'10: (5,7; 2,390)
f'11: (6,4; 2,535)
f'12: (7,1; 2,673)
f'13: (7,9; 2,803)
f'14: (8,6; 2,928)
f'15: (9,3; 3,047)
f'16: (10,0; 3,162)

Добавление точки (9, sqrt(9)):
Точки функции:
f'1: (0,0; 0,000)
f'2: (0,7; 0,845)
f'3: (1,4; 1,195)
f'4: (2,1; 1,464)
f'5: (2,9; 1,690)
```

```
f'6: (3,6; 1,890)
f'7: (4,3; 2,070)
f'8: (5,0; 2,236)
f'9: (5,6; 2,356)
f'10: (5,7; 2,390)
f'11: (6,4; 2,535)
f'12: (7,1; 2,673)
f'13: (7,9; 2,803)
f'14: (8,6; 2,928)
f'15: (9,0; 3,000)
f'16: (9,3; 3,047)
f'17: (10,0; 3,162)

Удаление точки с номером 5:
Точка f5 = (2,9, 1,690)
Точки функции:
f'1: (0,0; 0,000)
f'2: (0,7; 0,845)
f'3: (1,4; 1,195)
f'4: (2,1; 1,464)
f'5: (3,6; 1,890)
f'6: (4,3; 2,070)
f'7: (5,0; 2,236)
f'8: (5,6; 2,356)
f'9: (5,7; 2,390)
f'10: (6,4; 2,535)
f'11: (7,1; 2,673)
f'12: (7,9; 2,803)
f'13: (8,6; 2,928)
f'14: (9,0; 3,000)
f'15: (9,3; 3,047)
f'16: (10,0; 3,162)

Замена точки с номером 6 на точку f = (4,sqrt(4))
Исходная точка: f6'= (4,3; 2,070)
Измененная точка: f6'= (4,0; 2,000)
Точки функции:
f'1: (0,0; 0,000)
f'2: (0,7; 0,845)
```

```
f'3: (1,4; 1,195)
f'4: (2,1; 1,464)
f'5: (3,6; 1,890)
f'6: (4,0; 2,000)
f'7: (5,0; 2,236)
f'8: (5,6; 2,356)
f'9: (5,7; 2,390)
f'10: (6,4; 2,535)
f'11: (7,1; 2,673)
f'12: (7,9; 2,803)
f'13: (8,6; 2,928)
f'14: (9,0; 3,000)
f'15: (9,3; 3,047)
f'16: (10,0; 3,162)

Замена точки с номером 10 по значению x = 6
Исходная точка: f10'= (6,4; 2,535)
Измененная точка: f10'= (6,0; 2,449)
Точки функции:
f'1: (0,0; 0,000)
f'2: (0,7; 0,845)
f'3: (1,4; 1,195)
f'4: (2,1; 1,464)
f'5: (3,6; 1,890)
f'6: (4,0; 2,000)
f'7: (5,0; 2,236)
f'8: (5,6; 2,356)
f'9: (5,7; 2,390)
f'10: (6,0; 2,449)
f'11: (7,1; 2,673)
f'12: (7,9; 2,803)
f'13: (8,6; 2,928)
f'14: (9,0; 3,000)
f'15: (9,3; 3,047)
f'16: (10,0; 3,162)

Process finished with exit code 0
```