МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

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

ОТЧЕТ ПО
ЛАБОРАТОРНОЙ РАБОТЕ № 4

«Классы математических функций, ввод-вывод, сериализация»

по курсу
Объектно-ориентированное программирование

Выполнила: Егорова-Екимкова Яна,
студент группы 6203-010302D

Оглавление

Задание №1

Для задания 1 добавляю в классы ArrayTabulatedFunction и LinkedListTabulatedFunction конструкторы, которые получают все точки функции в виде массива.

```
public LinkedListTabulatedFunction(FunctionPoint[] points){  no usages  Егорова-Екимкова Яна *
    if(points.length < 2){
        throw new IllegalArgumentException("Points count must be more than two");
    }
    for (int i = points.length - 1; i > 0; i--){
        if (points[i].getX() < points[i-1].getX() - EPSILON){
            throw new IllegalArgumentException("Points must be in order");
        }
    }
    this.pointsCount = points.length;
    this.head = new FunctionNode(new FunctionPoint( x: 0, y: 0));
    head.next = head;
    head.prev = head;

    FunctionNode current = head;
    for (int i = 0; i < points.length; i++) {
        addNodeToTail(new FunctionPoint(points[i]));
    }

    this.lastIndex = 0;
    this.lastNode = head.next;
}
```

Рисунок 1

```
    public ArrayTabulatedFunction(FunctionPoint[] points){  3 usages  Егорова-Екимкова Яна
        if(points.length < 2){
            throw new IllegalArgumentException("Points count must be more than two");
        }
        for (int i = points.length - 1; i > 0; i--){
            if (points[i].getX() < points[i-1].getX() - EPSILON){
                throw new IllegalArgumentException("Points must be in order");
            }
        }
        this.pointsCount = points.length;
        this.array = new FunctionPoint[pointsCount];

        System.arraycopy(points,  srcPos: 0, this.array,  destPos: 0, pointsCount);
    }
```

Рисунок 2

Задание №2

Задание 2 требовало создать в пакете functions интерфейс Function и описать методы:

- public double getLeftDomainBorder() – возвращает значение левой границы области определения функции;
- public double getRightDomainBorder() – возвращает значение правой границы области определения функции;
- public double getFunctionValue(double x) – возвращает значение функции в заданной точкеРисунок 3

Затем необходимо было удалить описанные выше методы из интерфейса TabulatedFunction.

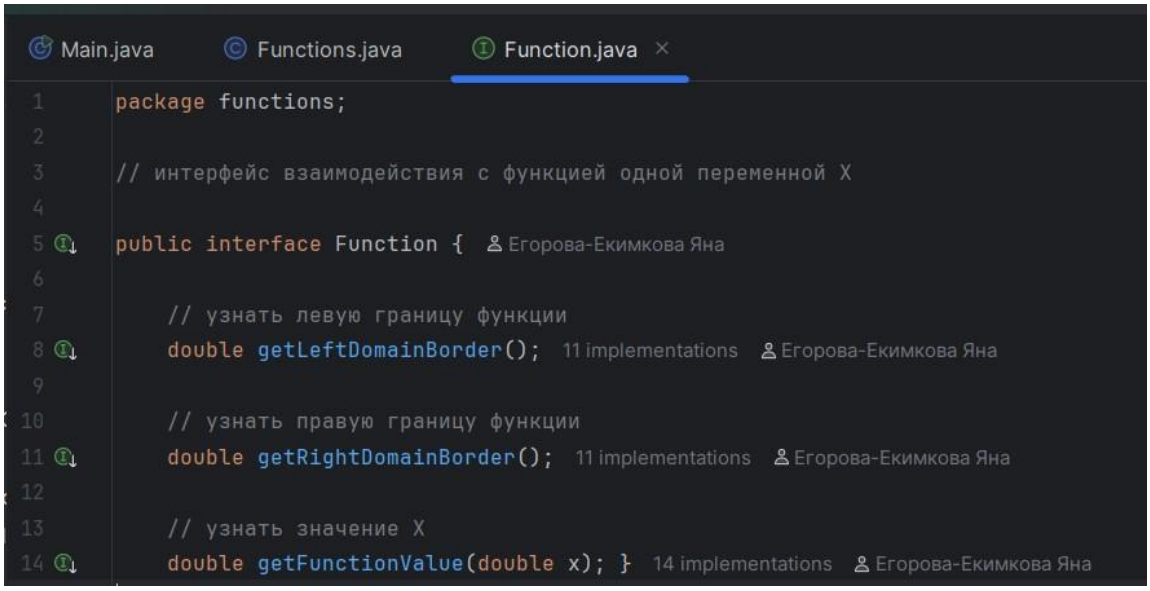


Рисунок 3

Задание №3

По заданию необходимо создать пакет functions.basic с описанными классами аналитически заданных функций:

- экспонента
- логарифм
- синус
- косинус
- тангенс

А также отдельный интерфейс для работы с тригонометрическими функциями.

```java
package functions.basic;

import functions.Function;

public class Exp implements Function {  2 usages  Егорова-Екимкова Яна
    @Override  Егорова-Екимкова Яна
    public double getLeftDomainBorder() { return Double.POSITIVE_INFINITY; }

    @Override  Егорова-Екимкова Яна
    public double getRightDomainBorder() { return Double.NEGATIVE_INFINITY; }

    @Override  Егорова-Екимкова Яна
    public double getFunctionValue(double x) { return Math.exp(x); }
}
```

Рисунок 4

```java
package functions.basic;

import functions.Function;

public class Log implements Function {  2 usages  Егорова-Екимкова Яна *
    double base;  2 usages

    public Log(double base) {  2 usages  Егорова-Екимкова Яна *
        if (base < 0 || base == 1) throw new IllegalArgumentException("Log base is incorrect");
        this.base = base;
    }

    @Override  Егорова-Екимкова Яна
    public double getLeftDomainBorder() { return 0; }

    @Override  Егорова-Екимкова Яна
    public double getRightDomainBorder() { return Double.POSITIVE_INFINITY; }

    @Override  Егорова-Екимкова Яна
    public double getFunctionValue(double x) {
        if (x <= 0) return Double.NaN;

        return Math.log(x) / Math.log(base);
    }
```

Рисунок 5

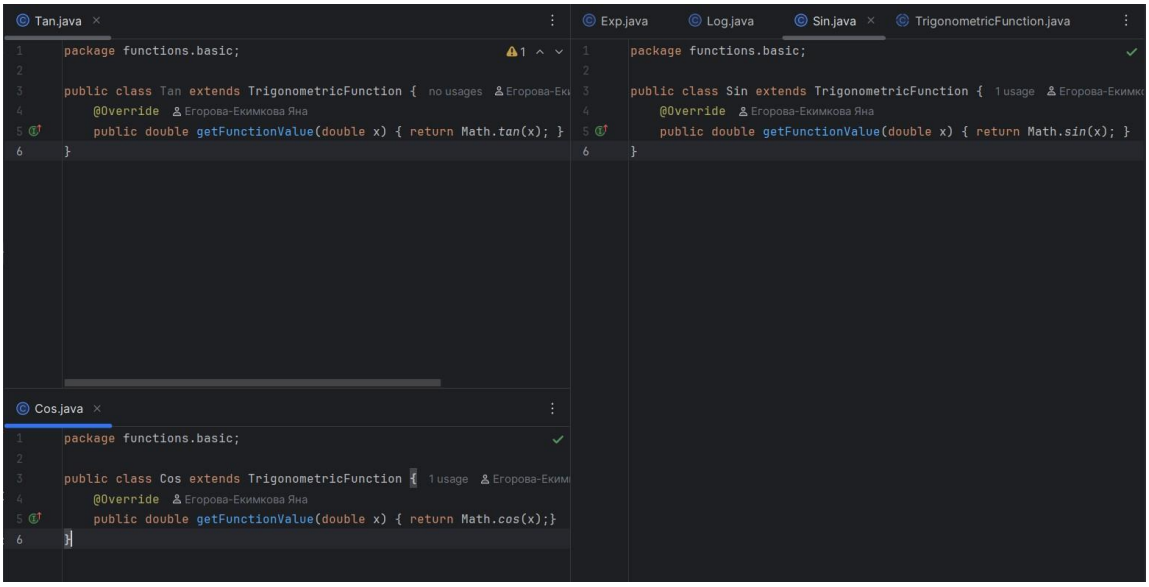


Рисунок 6

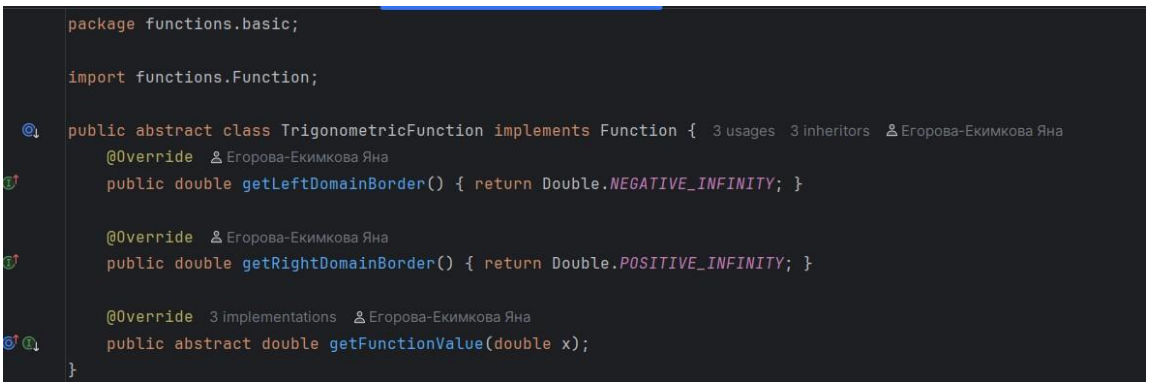


Рисунок 7

Задание №4

Для этого задания создаю пакет functions.meta для комбинированных функций и описываю классы:

- Сумма
- Умножение
- Возведение в степень
- Масштабирование
- Сдвиг
- Композиция

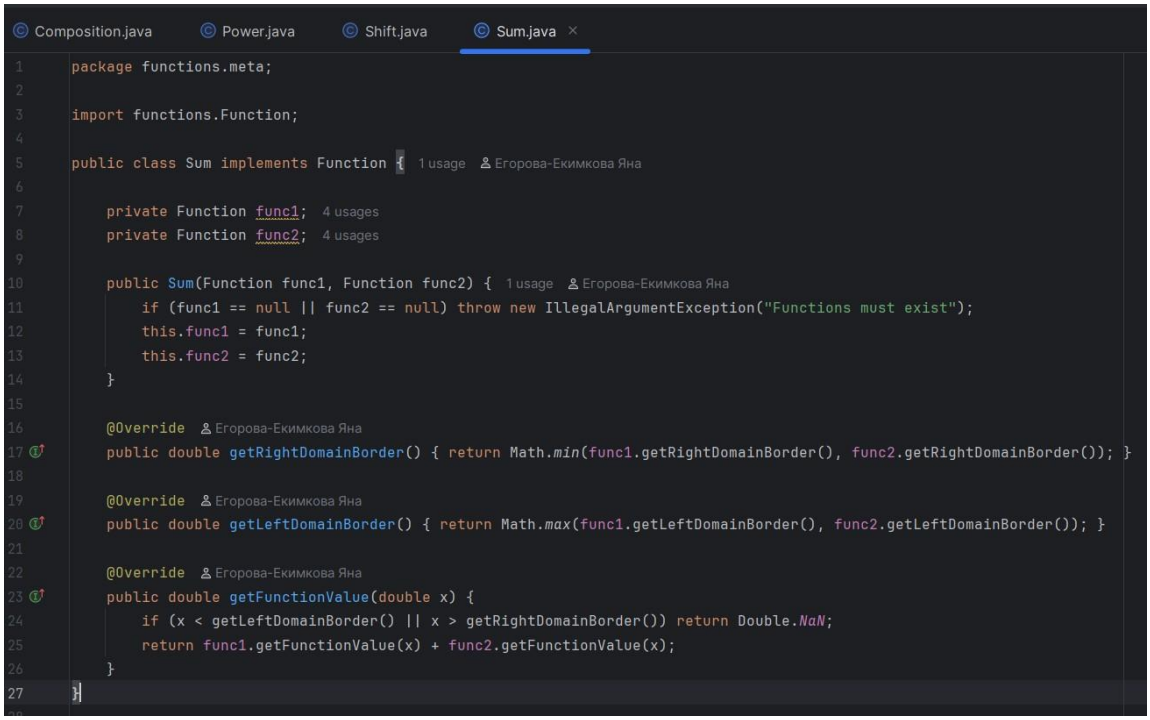


Рисунок 8

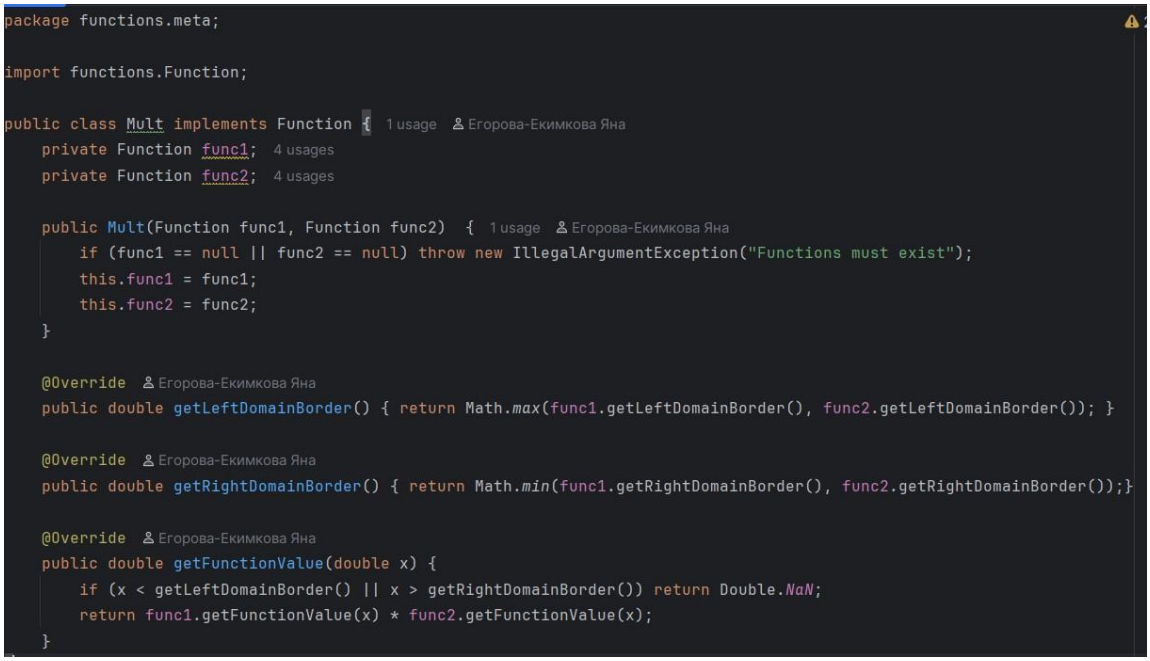
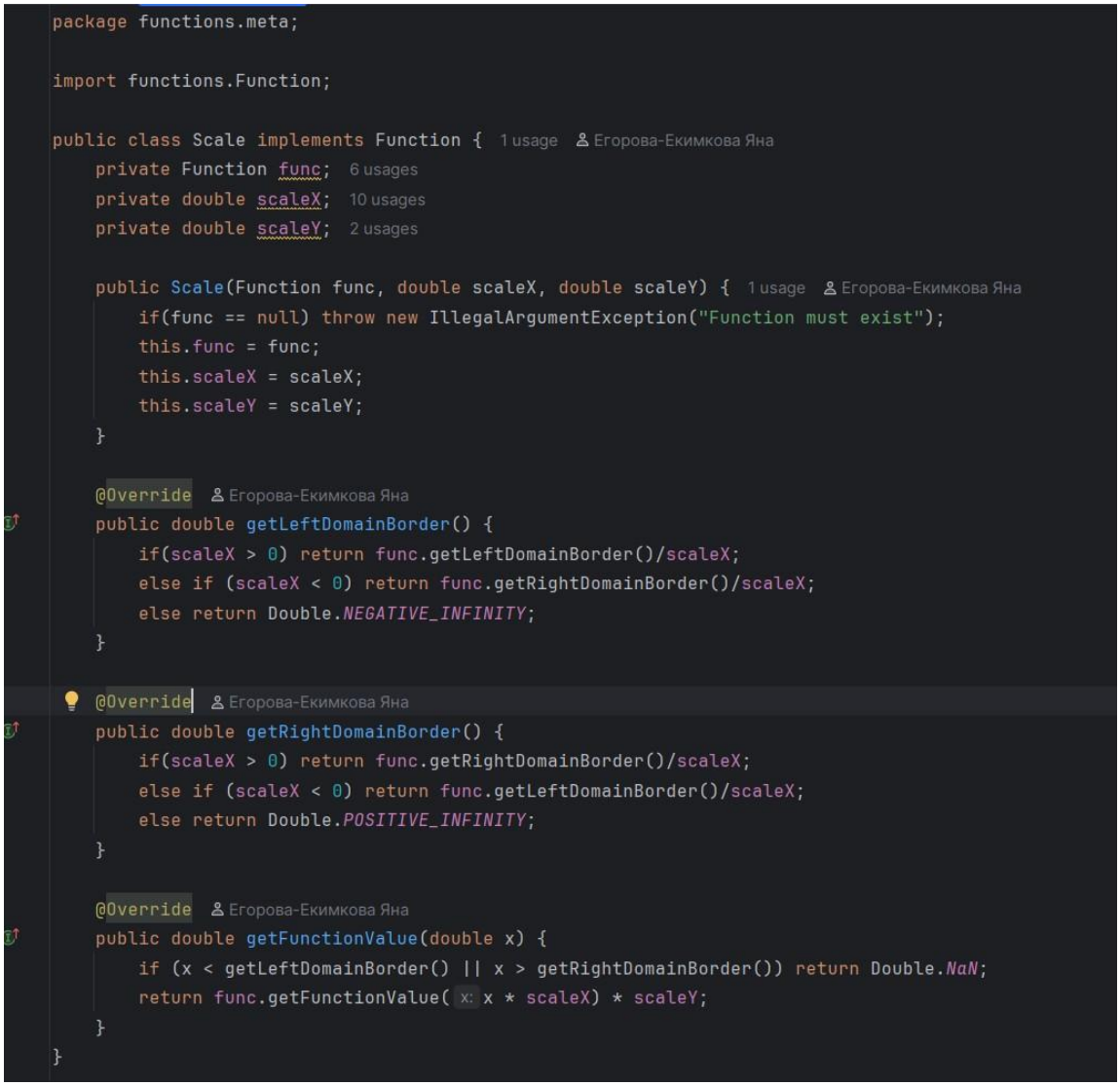


Рисунок 9

```java
package functions.meta;

import functions.Function;

public class Power implements Function {  1 usage  Егорова-Екимкова Яна
    private Function func;  4 usages
    private double pow;  2 usages

    public Power(Function func, double pow) {  1 usage  Егорова-Екимкова Яна
        if(func == null) throw new IllegalArgumentException("Function must exist");
        this.func = func;
        this.pow = pow;
    }

    @Override  Егорова-Екимкова Яна
    public double getRightDomainBorder() { return func.getRightDomainBorder(); }

    @Override  Егорова-Екимкова Яна
    public double getLeftDomainBorder() { return func.getLeftDomainBorder(); }

    @Override  Егорова-Екимкова Яна
    public double getFunctionValue(double x) {
        if (x < getLeftDomainBorder() || x > getRightDomainBorder()) return Double.NaN;
        return Math.pow(func.getFunctionValue(x), pow);
    }
}
```

Рисунок 10



```java
package functions.meta;

import functions.Function;

public class Scale implements Function {  1 usage  Егорова-Екимкова Яна
    private Function func;  6 usages
    private double scaleX;  10 usages
    private double scaleY;  2 usages

    public Scale(Function func, double scaleX, double scaleY) {  1 usage  Егорова-Екимкова Яна
        if(func == null) throw new IllegalArgumentException("Function must exist");
        this.func = func;
        this.scaleX = scaleX;
        this.scaleY = scaleY;
    }

    @Override  Егорова-Екимкова Яна
    public double getLeftDomainBorder() {
        if(scaleX > 0) return func.getLeftDomainBorder()/scaleX;
        else if (scaleX < 0) return func.getRightDomainBorder()/scaleX;
        else return Double.NEGATIVE_INFINITY;
    }

    @Override  Егорова-Екимкова Яна
    public double getRightDomainBorder() {
        if(scaleX > 0) return func.getRightDomainBorder()/scaleX;
        else if (scaleX < 0) return func.getLeftDomainBorder()/scaleX;
        else return Double.POSITIVE_INFINITY;
    }

    @Override  Егорова-Екимкова Яна
    public double getFunctionValue(double x) {
        if (x < getLeftDomainBorder() || x > getRightDomainBorder()) return Double.NaN;
        return func.getFunctionValue( x: x * scaleX) * scaleY;
    }
}
```

Рисунок 11

```java
package functions.meta;

import functions.Function;

public class Shift implements Function {  1 usage  Егорова-Екимкова Яна
    private Function func;  4 usages
    private double shiftX;  4 usages
    private double shiftY;  2 usages

    public Shift(Function func, double shiftX, double shiftY) {  1 usage  Егорова-Екимкова Яна
        if(func == null) throw new IllegalArgumentException("Function must exist");
        this.func = func;
        this.shiftX = shiftX;
        this.shiftY = shiftY;
    }

    @Override  Егорова-Екимкова Яна
    public double getRightDomainBorder() { return func.getRightDomainBorder() + shiftX; }

    @Override  Егорова-Екимкова Яна
    public double getLeftDomainBorder() { return func.getLeftDomainBorder() + shiftX; }

    @Override  Егорова-Екимкова Яна
    public double getFunctionValue(double x) {
        if (x < getLeftDomainBorder() || x > getRightDomainBorder()) return Double.NaN;
        return func.getFunctionValue( x: x - shiftX) + shiftY;
    }
}
```

Рисунок 12

```java
package functions.meta;

import functions.Function;

public class Composition implements Function {  1 usage  Егорова-Екимкова Яна
    private Function funcIn;  4 usages
    private Function funcOut;  2 usages

    public Composition(Function funcIn, Function funcOut) {  1 usage  Егорова-Екимкова Яна
        if (funcIn == null || funcOut == null) throw new IllegalArgumentException("Functions must exist");
        this.funcIn = funcIn;
        this.funcOut = funcOut;
    }

    @Override  Егорова-Екимкова Яна
    public double getRightDomainBorder() { return funcIn.getRightDomainBorder(); }

    @Override  Егорова-Екимкова Яна
    public double getLeftDomainBorder() { return funcIn.getLeftDomainBorder(); }

    @Override  Егорова-Екимкова Яна
    public double getFunctionValue(double x) {
        if (x < getLeftDomainBorder() || x > getRightDomainBorder()) return Double.NaN;
        return funcOut.getFunctionValue(funcIn.getFunctionValue(x));
    }
}
```

Рисунок 13

Задание №5

Для этого задания в пакете functions создаю интерфейс Functions для работы с функциями.

```java
package functions;
import functions.meta.*;

/*

вспомогательный класс, необходимый для проведения математических операций с различными видами функций и создания комбинированных функций,
не предусматривающий создание объектов типа данного класса

*/

public class Functions {  7 usages  Егорова-Екимкова Яна
    private Functions(){}  no usages  Егорова-Екимкова Яна
    // для того, чтобы возможности создания объекта вне класса не возникло, ограничиваем модификатор доступа конструктора на private

    // доступные математические операции

    // сумма функций
    public static Function sum(Function func1, Function func2) { return new Sum(func1, func2); }  2 usages  Егорова-Екимкова Яна

    // умножение функций
    public static Function mult(Function func1, Function func2) { return new Mult(func1, func2); }  no usages  Егорова-Екимкова Яна

    // возведение функции в степень
    public static Function power(Function func, double power) { return new Power(func, power); }  4 usages  Егорова-Екимкова Яна

    // масштабирование вдоль осей XoY
    public static Function scale(Function func, double scaleX, double scaleY) { return new Scale(func, scaleX, scaleY); }  no usages  Егорова-Екимкова Яна

    // сдвиг осей XoY
    public static Function shift(Function func, double shiftX, double shiftY) { return new Shift(func, shiftX, shiftY); }  no usages  Егорова-Екимкова Яна

    // композиция двух функций
    public static Function composition(Function func1, Function func2) { return new Composition(func1, func2); }  1 usage  Егорова-Екимкова Яна

}
```

Рисунок 14

Задание №6

В пакете functions создала класс TabulatedFunctions для работы с табулированными функциями.

```java
package functions;

import java.io.*;

public class TabulatedFunctions {  11 usages  Егорова-Екимкова Яна *

    private TabulatedFunctions(){}; // приватный конструктор для исключения возможности создания объекта данного кла

    public static TabulatedFunction tabulate(Function func, double leftX, double rightX, int pointsCount){  7 usages
        if (func == null) throw new IllegalArgumentException("Function must exist");
        if (leftX > rightX) throw new IllegalArgumentException("Left border must be less than right border");
        if (pointsCount < 2) throw new IllegalArgumentException("Points count must be more than 2");

        FunctionPoint point[] = new FunctionPoint[pointsCount];
        double step = (rightX - leftX) / (pointsCount - 1);
        for (int i = 0; i<pointsCount; i++){
            double x = leftX + i*step;
            double y = func.getFunctionValue(x);
            point[i] = new FunctionPoint(x,y);
        }
        return new ArrayTabulatedFunction(point);
    }
```

Рисунок 15

Задание №7

В классе TabulatedFunctions были реализованы дополнительные методы для работы с потоками ввода-вывода.

Все методы объявлены с throws IOException, передавая обработку исключений вызывающему коду. Такое решение позволяет внешнему коду самостоятельно управлять реакцией на ошибки ввода-вывода.

Потоки не закрываются внутри методов, чтобы не нарушать их возможное дальнейшее использование. Управление ресурсами оставлено на усмотрение вызывающей стороны.

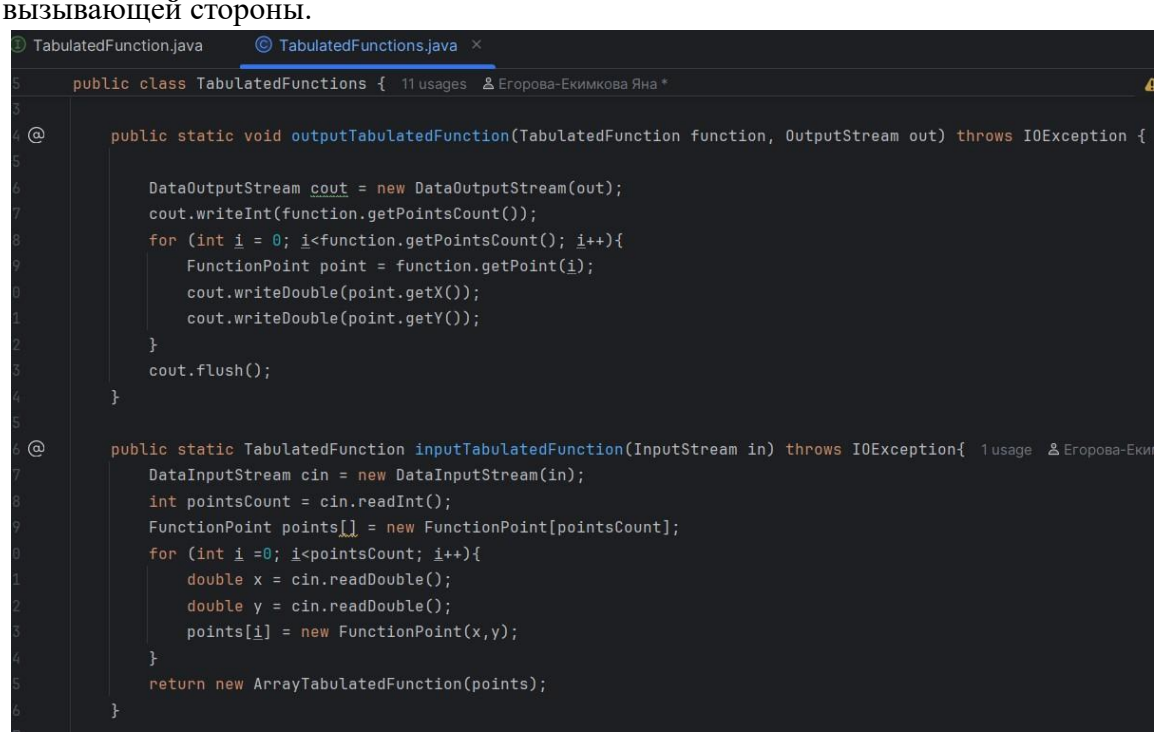


Рисунок 16

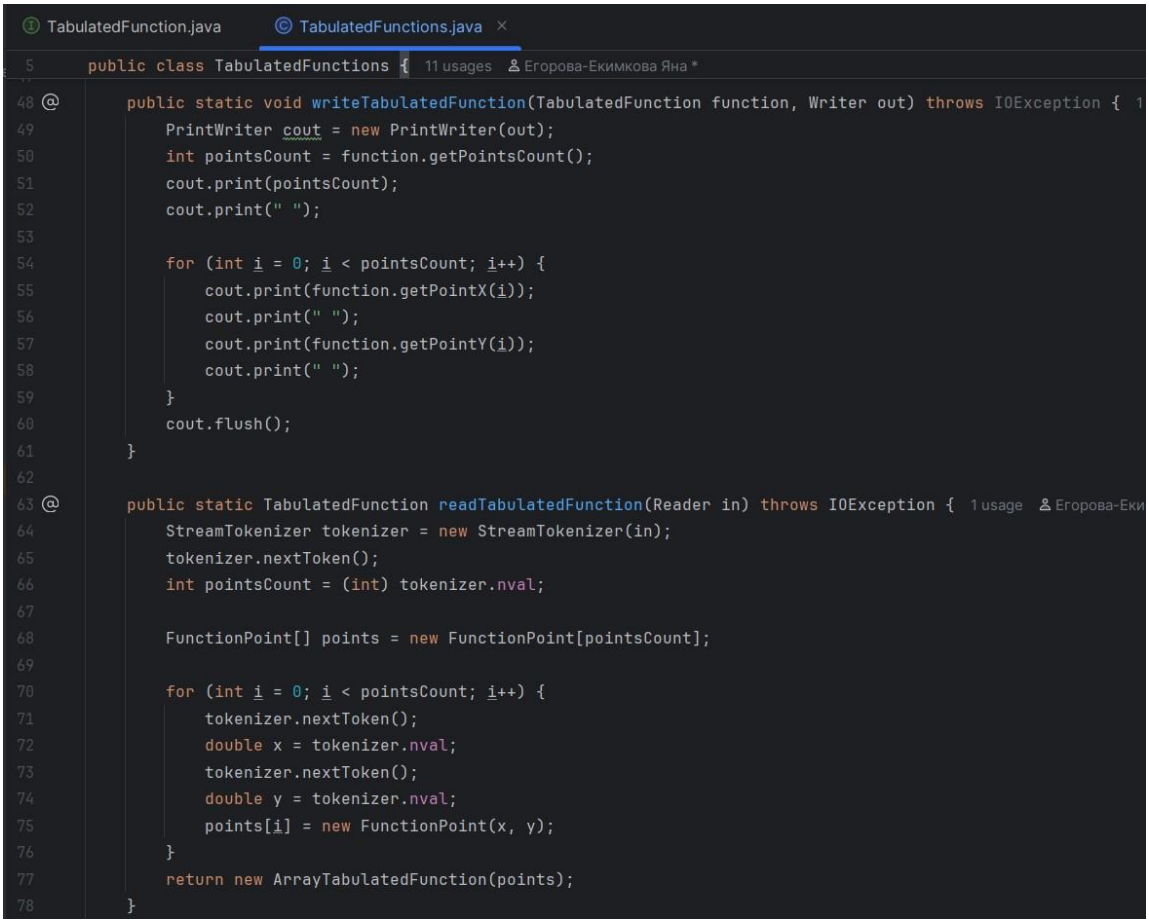


Рисунок 17

Задание №8

Проверяю работу описанных ранее классов. Синус и косинус.

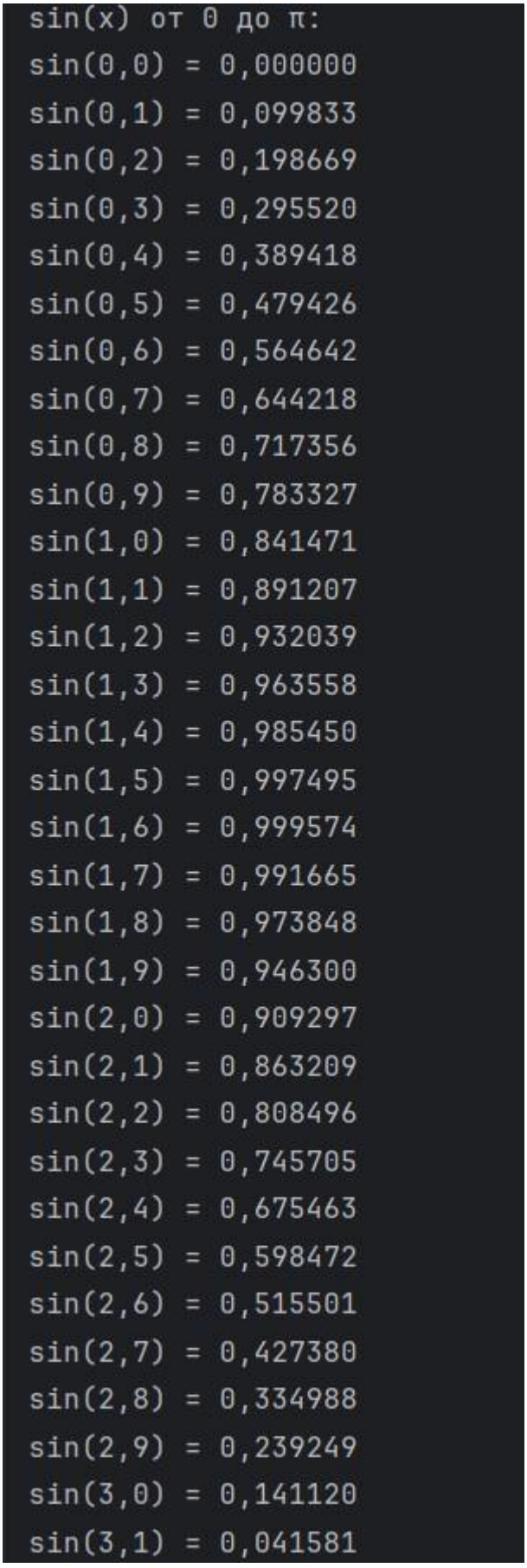

sin(x) от 0 до π:
sin(0,0) = 0,000000
sin(0,1) = 0,099833
sin(0,2) = 0,198669
sin(0,3) = 0,295520
sin(0,4) = 0,389418
sin(0,5) = 0,479426
sin(0,6) = 0,564642
sin(0,7) = 0,644218
sin(0,8) = 0,717356
sin(0,9) = 0,783327
sin(1,0) = 0,841471
sin(1,1) = 0,891207
sin(1,2) = 0,932039
sin(1,3) = 0,963558
sin(1,4) = 0,985450
sin(1,5) = 0,997495
sin(1,6) = 0,999574
sin(1,7) = 0,991665
sin(1,8) = 0,973848
sin(1,9) = 0,946300
sin(2,0) = 0,909297
sin(2,1) = 0,863209
sin(2,2) = 0,808496
sin(2,3) = 0,745705
sin(2,4) = 0,675463
sin(2,5) = 0,598472
sin(2,6) = 0,515501
sin(2,7) = 0,427380
sin(2,8) = 0,334988
sin(2,9) = 0,239249
sin(3,0) = 0,141120
sin(3,1) = 0,041581

Рисунок 18

```
cos(x) от 0 до π:
cos(0,0) = 1,000000
cos(0,1) = 0,995004
cos(0,2) = 0,980067
cos(0,3) = 0,955336
cos(0,4) = 0,921061
cos(0,5) = 0,877583
cos(0,6) = 0,825336
cos(0,7) = 0,764842
cos(0,8) = 0,696707
cos(0,9) = 0,621610
cos(1,0) = 0,540302
cos(1,1) = 0,453596
cos(1,2) = 0,362358
cos(1,3) = 0,267499
cos(1,4) = 0,169967
cos(1,5) = 0,070737
cos(1,6) = -0,029200
cos(1,7) = -0,128844
cos(1,8) = -0,227202
cos(1,9) = -0,323290
cos(2,0) = -0,416147
cos(2,1) = -0,504846
cos(2,2) = -0,588501
cos(2,3) = -0,666276
cos(2,4) = -0,737394
cos(2,5) = -0,801144
cos(2,6) = -0,856889
cos(2,7) = -0,904072
cos(2,8) = -0,942222
cos(2,9) = -0,970958
cos(3,0) = -0,989992
cos(3,1) = -0,999135
```

Рисунок 19

Табулированные синус и косинус, сравнение с оригиналами.

```
Сравнение точных и табулированных значений:
x=0,0: sin=0,000000(tab=0,000000) cos=1,000000(tab=1,000000)
x=0,1: sin=0,099833(tab=0,097982) cos=0,995004(tab=0,982723)
x=0,2: sin=0,198669(tab=0,195963) cos=0,980067(tab=0,965446)
x=0,3: sin=0,295520(tab=0,293945) cos=0,955336(tab=0,948170)
x=0,4: sin=0,389418(tab=0,385907) cos=0,921061(tab=0,914355)
x=0,5: sin=0,479426(tab=0,472070) cos=0,877583(tab=0,864608)
x=0,6: sin=0,564642(tab=0,558234) cos=0,825336(tab=0,814862)
x=0,7: sin=0,644218(tab=0,643982) cos=0,764842(tab=0,764620)
x=0,8: sin=0,717356(tab=0,707935) cos=0,696707(tab=0,688404)
x=0,9: sin=0,783327(tab=0,771888) cos=0,621610(tab=0,612188)
x=1,0: sin=0,841471(tab=0,835841) cos=0,540302(tab=0,535972)
x=1,1: sin=0,891207(tab=0,883993) cos=0,453596(tab=0,450633)
x=1,2: sin=0,932039(tab=0,918022) cos=0,362358(tab=0,357141)
x=1,3: sin=0,963558(tab=0,952051) cos=0,267499(tab=0,263648)
x=1,4: sin=0,985450(tab=0,984808) cos=0,169967(tab=0,169931)
x=1,5: sin=0,997495(tab=0,984808) cos=0,070737(tab=0,070437)
x=1,6: sin=0,999574(tab=0,984808) cos=-0,029200(tab=-0,029056)
x=1,7: sin=0,991665(tab=0,984808) cos=-0,128844(tab=-0,128549)
x=1,8: sin=0,973848(tab=0,966204) cos=-0,227202(tab=-0,224761)
x=1,9: sin=0,946300(tab=0,932175) cos=-0,323290(tab=-0,318254)
x=2,0: sin=0,909297(tab=0,898147) cos=-0,416147(tab=-0,411747)
x=2,1: sin=0,863209(tab=0,862441) cos=-0,504846(tab=-0,504272)
x=2,2: sin=0,808496(tab=0,798488) cos=-0,588501(tab=-0,580488)
x=2,3: sin=0,745705(tab=0,734535) cos=-0,666276(tab=-0,656704)
x=2,4: sin=0,675463(tab=0,670582) cos=-0,737394(tab=-0,732920)
x=2,5: sin=0,598472(tab=0,594072) cos=-0,801144(tab=-0,794171)
x=2,6: sin=0,515501(tab=0,507908) cos=-0,856889(tab=-0,843917)
x=2,7: sin=0,427380(tab=0,421745) cos=-0,904072(tab=-0,893664)
x=2,8: sin=0,334988(tab=0,334698) cos=-0,942222(tab=-0,940984)
x=2,9: sin=0,239249(tab=0,236716) cos=-0,970958(tab=-0,958261)
x=3,0: sin=0,141120(tab=0,138735) cos=-0,989992(tab=-0,975537)
x=3,1: sin=0,041581(tab=0,040753) cos=-0,999135(tab=-0,992814)
```

Рисунок 20 sin^2(x)+

cos^2(x) = 1

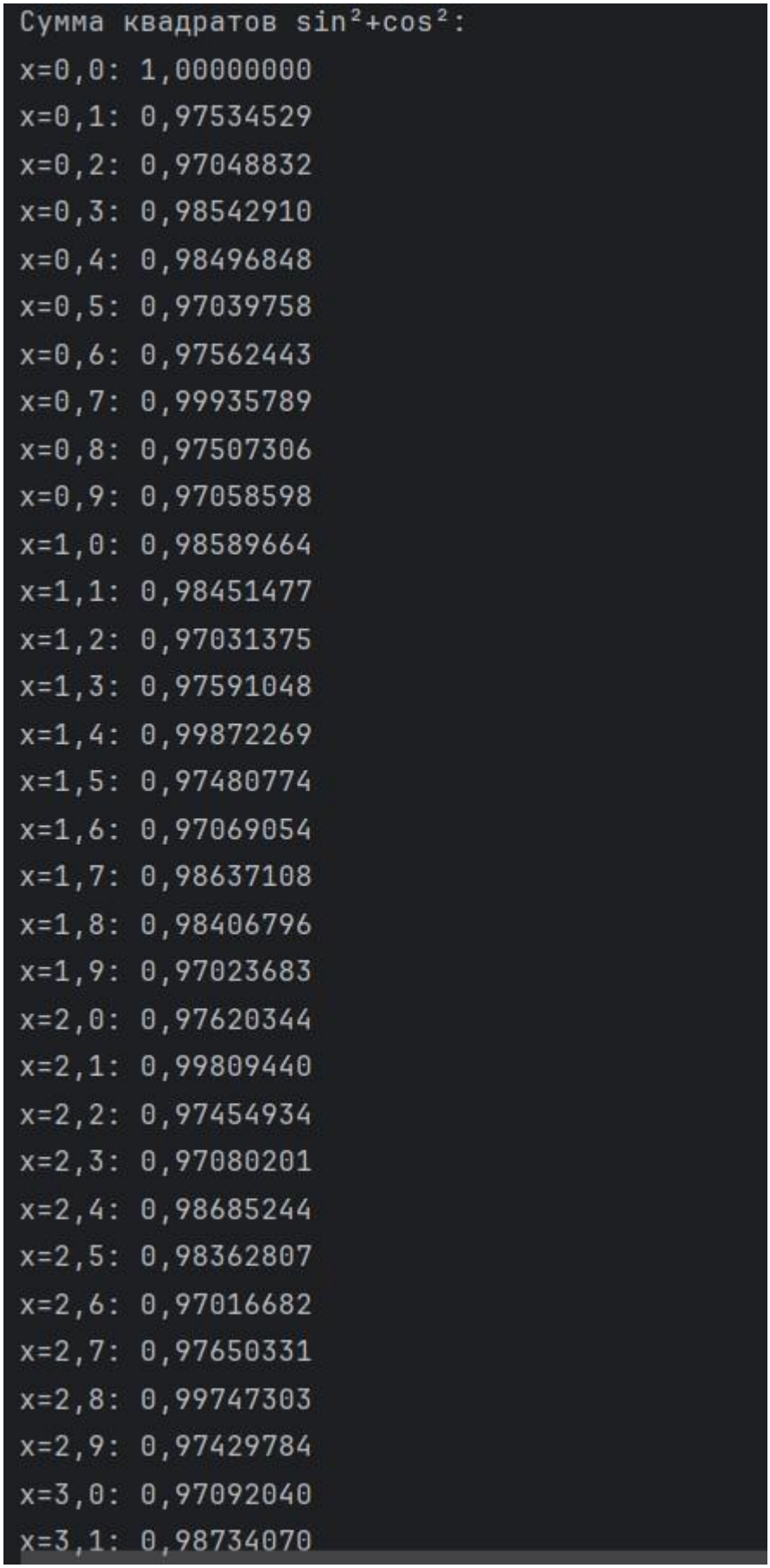

Сумма квадратов sin²+cos²:
x=0,0: 1,00000000
x=0,1: 0,97534529
x=0,2: 0,97048832
x=0,3: 0,98542910
x=0,4: 0,98496848
x=0,5: 0,97039758
x=0,6: 0,97562443
x=0,7: 0,99935789
x=0,8: 0,97507306
x=0,9: 0,97058598
x=1,0: 0,98589664
x=1,1: 0,98451477
x=1,2: 0,97031375
x=1,3: 0,97591048
x=1,4: 0,99872269
x=1,5: 0,97480774
x=1,6: 0,97069054
x=1,7: 0,98637108
x=1,8: 0,98406796
x=1,9: 0,97023683
x=2,0: 0,97620344
x=2,1: 0,99809440
x=2,2: 0,97454934
x=2,3: 0,97080201
x=2,4: 0,98685244
x=2,5: 0,98362807
x=2,6: 0,97016682
x=2,7: 0,97650331
x=2,8: 0,99747303
x=2,9: 0,97429784
x=3,0: 0,97092040
x=3,1: 0,98734070

Рисунок 21

Различное количество точек.

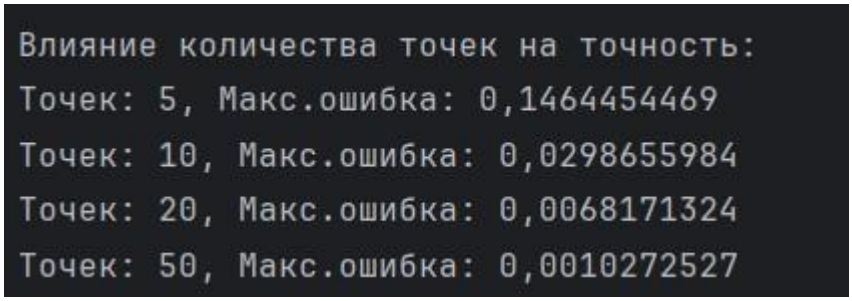


Рисунок 22

Экспонента, логарифм и работа с файлом, сравнение.

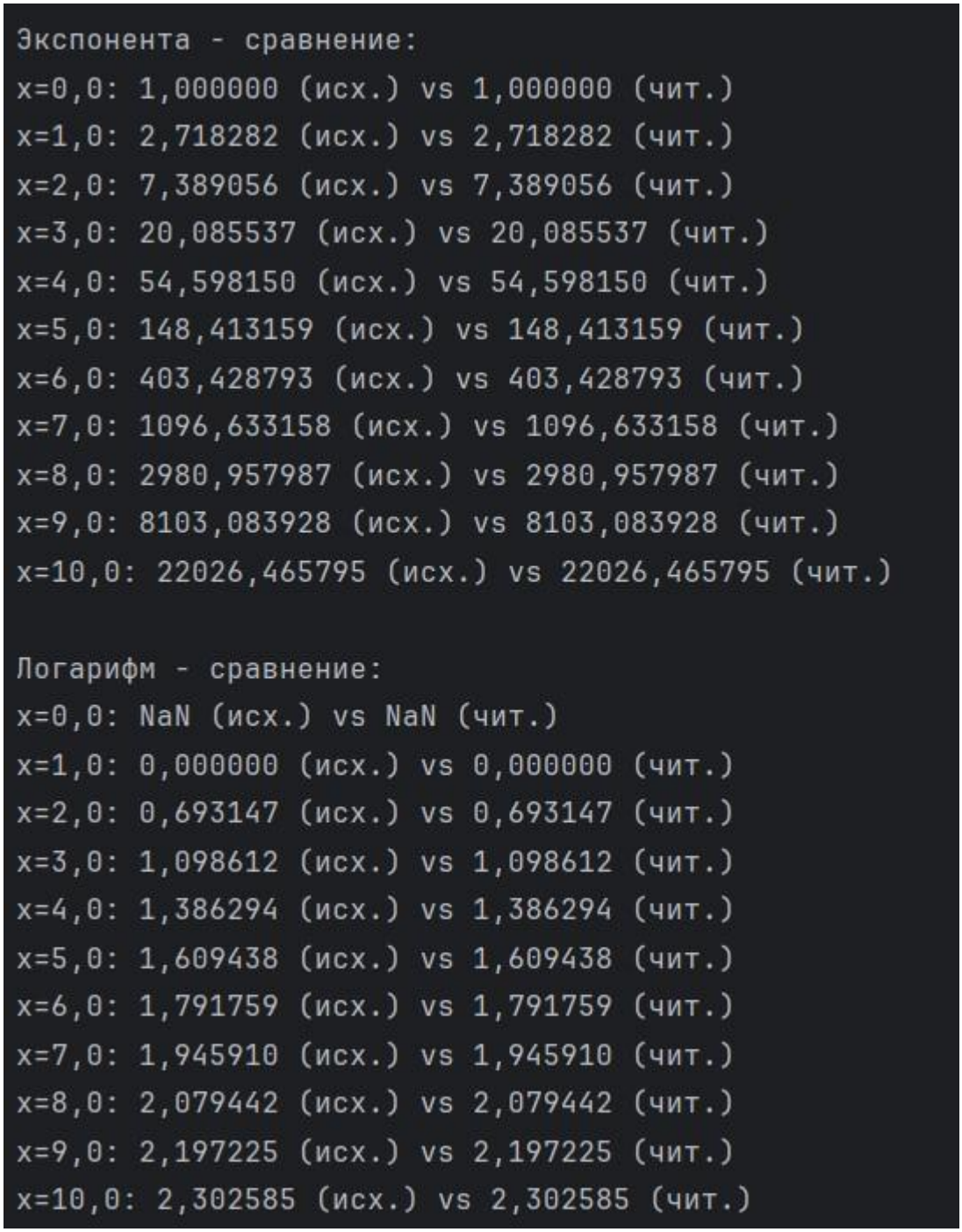


Рисунок 23

Вывод: Бинарные файлы занимают меньше места и обрабатываются быстрее. Текстовые файлы можно открыть и прочитать в любом редакторе, что удобно для проверки.

Задание №9

Для выполнения задания я ознакомилась с интерфейсами java.io.Serializable и java.io.Externalizable. Serializable автоматически сохраняет и восстанавливает объекты, что удобно, но может работать медленнее и создавать файлы большего размера.
Externalizable требует ручного написания кода для сериализации, но даёт больше контроля над процессом и позволяет оптимизировать результат.

```
@Override  Егорова-Екимкова Яна
public void writeExternal(ObjectOutput out) throws IOException {
    out.writeInt(pointsCount);

    for (int i = 0; i < pointsCount; i++) {
        out.writeDouble(array[i].getX());
        out.writeDouble(array[i].getY());
    }
}

@Override  Егорова-Екимкова Яна
public void readExternal(ObjectInput in) throws IOException, ClassNotFoundException {
    pointsCount = in.readInt();
    array = new FunctionPoint[pointsCount];

    for (int i = 0; i < pointsCount; i++) {
        double x = in.readDouble();
        double y = in.readDouble();
        array[i] = new FunctionPoint(x, y);
    }
}
```

Рисунок 24

```java
@Override  Егорова-Екимкова Яна
public void writeExternal(ObjectOutput out) throws IOException {
    out.writeInt(pointsCount);
    FunctionNode current = head.next;

    while (current != head) {
        out.writeDouble(current.point.getX());
        out.writeDouble(current.point.getY());
        current = current.next;
    }
}

@Override  Егорова-Екимкова Яна
public void readExternal(ObjectInput in) throws IOException, ClassNotFoundException {
    head.next = head;
    head.prev = head;
    pointsCount = 0;

    int pointsCount = in.readInt();

    for (int i = 0; i < pointsCount; i++)
    {
        double x = in.readDouble();
        double y = in.readDouble();
        addNodeToTail(new FunctionPoint(x, y));
    }
}
```

Рисунок 25

```java
//интерфейс взаимодействия с табулированной функцией (двусвязный список или массив)

public interface TabulatedFunction extends Function, Externalizable {  18 usages  2 implementations
```

Рисунок 26

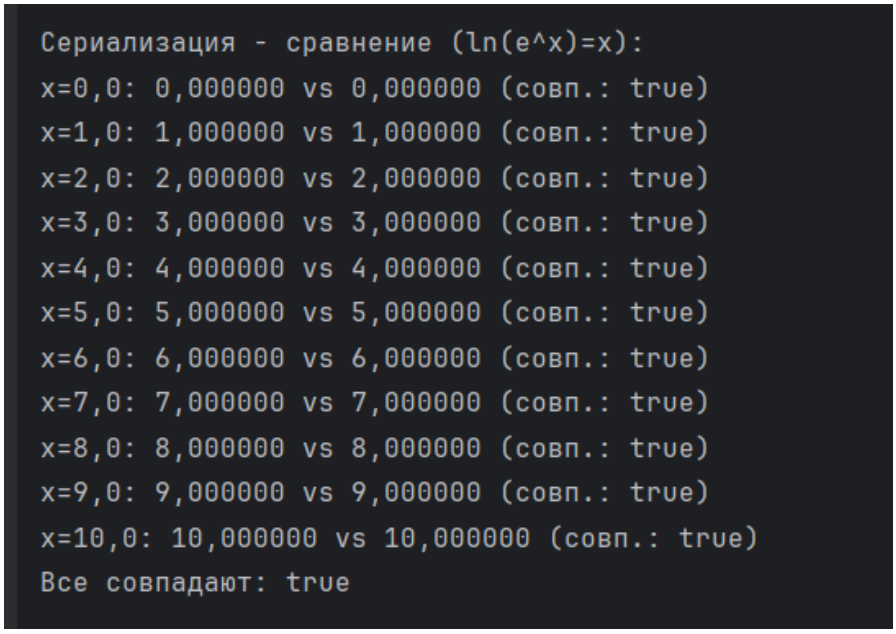

Сериализация - сравнение (ln(e^x)=x):
x=0,0: 0,000000 vs 0,000000 (совп.: true)
x=1,0: 1,000000 vs 1,000000 (совп.: true)
x=2,0: 2,000000 vs 2,000000 (совп.: true)
x=3,0: 3,000000 vs 3,000000 (совп.: true)
x=4,0: 4,000000 vs 4,000000 (совп.: true)
x=5,0: 5,000000 vs 5,000000 (совп.: true)
x=6,0: 6,000000 vs 6,000000 (совп.: true)
x=7,0: 7,000000 vs 7,000000 (совп.: true)
x=8,0: 8,000000 vs 8,000000 (совп.: true)
x=9,0: 9,000000 vs 9,000000 (совп.: true)
x=10,0: 10,000000 vs 10,000000 (совп.: true)
Все совпадают: true

Рисунок 27