Харківський національний університет імені В. Н. Каразіна
Факультет комп'ютерних наук
Кафедра штучного інтелекту та програмного забезпечення

ЗВІТ
З РОЗРАХУНКОВО-ГРАФІЧНОЇ РОБОТИ №1

дисципліна: «Крос-платформне
програмування»

Виконав: студент групи КС23
Травченко Сергій Миколайович

Перевірив: викладач кафедри ШІтаПЗ
Споров Олександр Євгенович

Харків
2024

«Згадати все…»

Розглянемо поняття функції — в математиці це відповідність між елементами двох множин, що встановлена за таким правилом: кожному елементу першої множини відповідає один і лише один елемент другої множини. Математичне поняття функції виражає те, яким чином одна величина повністю визначає значення іншої величини. Задати функцію означає встановити правило, за допомогою якого за заданими значеннями незалежної змінної можна знайти відповідні значення функції. Розглянемо лише два способи задання функцій:

- Аналітичний спосіб. Закон, що встановлює зв'язок між аргументом та функцією, визначається за допомогою формул.
- Табличний спосіб. Закон, що встановлює зв'язок між аргументом та функцією, задається за допомогою таблиці, де вказані значення аргументу та відповідні їм значення функції.
- Завдання: створити програмне рішення, що моделює поняття функції та дозволяє:
- створювати функції, що задані або аналітично (вираз в коді), або ж таблично;
- обчислювати значення функції для вказаного аргументу;
- числовим методом із вказаною точністю обчислювати похідну заданої функції.

Завдання №1 Створити програмне рішення для одноманітного представлення функції однієї змінної (тип double), заданої різними способами: аналітичним (формула, що визначає функцію, вказана в тексті програми) та табличним. Чисельно продиференціювати із зазначеною точністю задані таким чином функції однієї змінної. Для розв’язку задачі скористатись рекомендаціями, що наведені нижче, та створити програмне рішення за наведеною в рекомендаціях схемою.

Для тестування програмного рішення обчисліть із заданою точністю на відрізку $x \in [1.5 , 6.5]$ з кроком 0.05 для декількох тестових функцій значення функції та її похідної. Результати диференціювання зберегти в текстових файлах даних. Графічно відобразіть збережені дані за допомогою інших стандартних програмних рішень. Для тестування програми використати такі функції:

- перша функція: $f(x)=\exp(-x^2)\cdot\sin(x)$;
- друга функція: $f(x)=\exp(-a\,x^2)\cdot\sin(x)$ для заданих значень a ($a=0.5 ,1.0 ,1.5$);
- третя функція: функція задана табличним чином — значення незалежної змінної та — 1 — функції (sin(x)) збережені у текстовому файлі даних. Таблиця значень функції зберігається у відповідному класі у вигляді змінної типу ArrayList

Результати виконання завдання №1 наведено:

1. У лістингу 1 – вихідний код програми;
2. На малюнку 1.1 – результат виконання програми;
3. На таблиці 1 – графічний результат обчислень.

Лістинг 1. Вихідний код програми

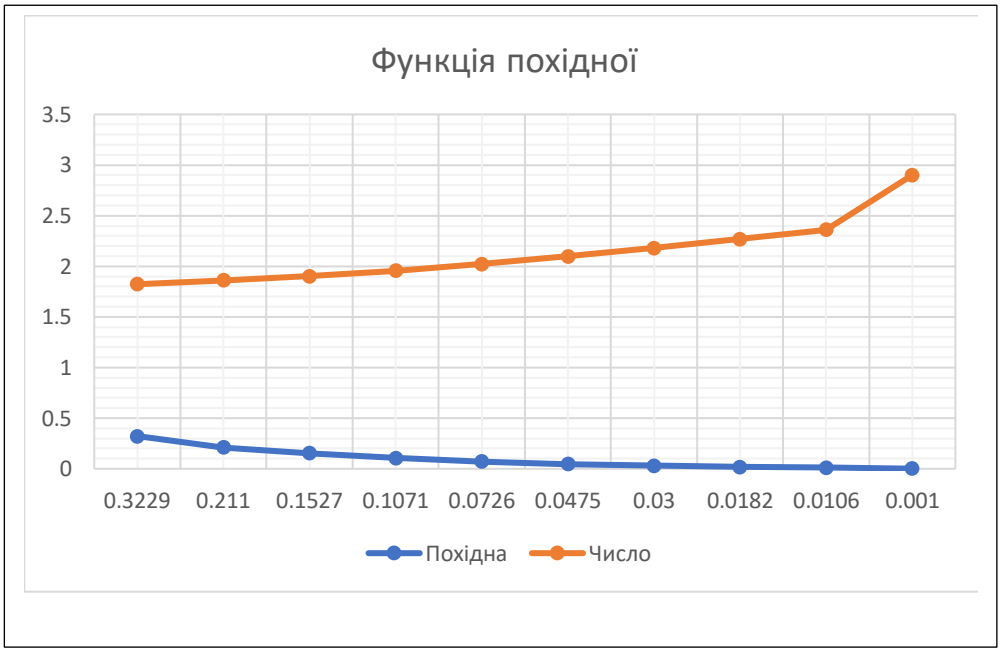
```java
import java.util.ArrayList;

// Клас для представлення аналітичної функції та обчислення похідних
2 usages
class AnalyticalFunction {
    // Представлення функції
    1 usage
    static double function(double x) { return Math.exp(-x*x) * Math.sin(x); }

    // Обчислення похідної
    1 usage
    static double derivative(double x) { return Math.exp(-x*x) * (2*x*Math.sin(x) + Math.cos(x)); }
}

// Клас для представлення табличної функції та обчислення похідних
2 usages
class TabularFunction {
    // Таблиця значень функції
    6 usages
    private ArrayList<Double> xValues;
    3 usages
    private ArrayList<Double> yValues;

    // Конструктор для ініціалізації таблиці значень
    1 usage
    public TabularFunction(ArrayList<Double> xValues, ArrayList<Double> yValues) {
        this.xValues = xValues;
        this.yValues = yValues;
    }

    // Представлення функції
    3 usages
```

```java
    1 usage
    private static void testAnalyticalFunction() {
        System.out.println("Analytical Function Test:");
        for (double x = 1.5; x <= 6.5; x += 0.05) {
            double y = AnalyticalFunction.function(x);
            double dy = AnalyticalFunction.derivative(x);
            System.out.printf("x=%.2f, f(x)=%.4f, f'(x)=%.4f\n", x, y, dy);
        }
    }

    1 usage
    private static void testTabularFunction() {
        // Приклад табличної функції
        ArrayList<Double> xValues = new ArrayList<>();
        ArrayList<Double> yValues = new ArrayList<>();

        // Додавання значень до таблиці
        for (double x = 1.5; x <= 6.5; x += 0.1) {
            xValues.add(x);
            yValues.add(Math.sin(x));
        }

        TabularFunction tabularFunction = new TabularFunction(xValues, yValues);

        System.out.println("\nTabular Function Test:");
        for (double x = 1.5; x <= 6.5; x += 0.05) {
            double y = tabularFunction.function(x);
            double dy = tabularFunction.derivative(x);
            System.out.printf("x=%.2f, f(x)=%.4f, f'(x)=%.4f\n", x, y, dy);
        }
    }
```

```
D:\Java\bin\java.exe "-javaagent:C:\Program
Analytical Function Test:
x=1,50, f(x)=0,1051, f'(x)=0,3229
x=1,55, f(x)=0,0905, f'(x)=0,2823
x=1,60, f(x)=0,0773, f'(x)=0,2450
x=1,65, f(x)=0,0655, f'(x)=0,2110
x=1,70, f(x)=0,0551, f'(x)=0,1802
x=1,75, f(x)=0,0460, f'(x)=0,1527
x=1,80, f(x)=0,0381, f'(x)=0,1284
x=1,85, f(x)=0,0314, f'(x)=0,1071
x=1,90, f(x)=0,0256, f'(x)=0,0885
x=1,95, f(x)=0,0207, f'(x)=0,0726
x=2,00, f(x)=0,0167, f'(x)=0,0590
x=2,05, f(x)=0,0133, f'(x)=0,0475
x=2,10, f(x)=0,0105, f'(x)=0,0379
x=2,15, f(x)=0,0082, f'(x)=0,0300
x=2,20, f(x)=0,0064, f'(x)=0,0235
x=2,25, f(x)=0,0049, f'(x)=0,0182
x=2,30, f(x)=0,0038, f'(x)=0,0139
x=2,35, f(x)=0,0028, f'(x)=0,0106
x=2,40, f(x)=0,0021, f'(x)=0,0079
x=2,45, f(x)=0,0016, f'(x)=0,0058
x=2,50, f(x)=0,0012, f'(x)=0,0042
x=2,55, f(x)=0,0008, f'(x)=0,0030
x=2,60, f(x)=0,0006, f'(x)=0,0021
x=2,65, f(x)=0,0004, f'(x)=0,0014
```

```
x=5,65, f(x)=-0,5910, f'(x)=-136,7234
x=5,70, f(x)=-0,5507, f'(x)=0,8608
x=5,75, f(x)=-0,5076, f'(x)=-115,1697
x=5,80, f(x)=-0,4646, f'(x)=0,9073
x=5,85, f(x)=-0,4192, f'(x)=-92,4654
x=5,90, f(x)=-0,3739, f'(x)=0,9446
x=5,95, f(x)=-0,3266, f'(x)=-68,8371
x=6,00, f(x)=-0,2794, f'(x)=0,9725
x=6,05, f(x)=-0,2308, f'(x)=-44,5211
x=6,10, f(x)=-0,1822, f'(x)=0,9907
x=6,15, f(x)=-0,1326, f'(x)=-19,7602
x=6,20, f(x)=-0,0831, f'(x)=0,9990
x=6,25, f(x)=-0,0331, f'(x)=5,1982
x=6,30, f(x)=0,0168, f'(x)=0,9974
x=6,35, f(x)=0,0667, f'(x)=30,1046
x=6,40, f(x)=0,1165, f'(x)=0,9857
Exception in thread "main" java.lang.IndexOutOfBoundsException Create breakpoint : Index: 51, Size: 51
    at java.util.ArrayList.rangeCheck(ArrayList.java:659)
    at java.util.ArrayList.get(ArrayList.java:435)
    at TabularFunction.function(Main.java:34)
    at TabularFunction.derivative(Main.java:45)
    at Main.testTabularFunction(Main.java:100)
    at Main.main(Main.java:72)

Process finished with exit code 1
```

Малюнок 1.1 – результат виконання програми



Таблиця 1 – графічний результат обчислень, створений за допомогою Microsoft Excel

- Як бачимо, ми зіштовхнулися з проблемою, яка невід'ємно пов'язана з програмуванням, а саме з обчисленням функції, чим довше вона працює, тим помилковіше становиться результат, що видно на таблиці.

Завдання №2 Провести рефакторинг створеної системи. (Рефакторинг (англ. refactoring) — процес редагування програмного коду, внутрішньої структури програмного забезпечення для полегшення розуміння коду та внесення подальших правок без зміни зовнішньої поведінки самої системи.)
Для цього проаналізувати створене програмне рішення, визначити недоліки його структури з точки зору об'єктно-орієнтованого підходу та вирішити знайдені проблеми, змінивши структуру програмного рішення (наприклад, прибрати статичні методи (за винятком методу main()), покращити структуру класів, прибравши зайві класи, непотрібне наслідування, зробити структуру більш зручною до подальшого розширення функціональності і т. і.). Для цього рекомендується згадати основні концепції об'єктно-орієнтованого підходу, принципи SOLID (англ. Single responsibility principle, Open/closed principle, Liskov substitution principle, Interface segregation principle, Dependency inversion principle) та набір патернів GRASP (англ. General Responsibility Assignment Software Patterns). Протестувати створене програмне рішення на попередній тестових випадках.

- Результати виконання завдання 2 – На мою думку, ще на початку написання програми я дотримувався правил чистого програмування, і проблема розрахунку не в неграмотному написанні коду, а у операційній системі та системі обчислення самого середовища та компілятора.

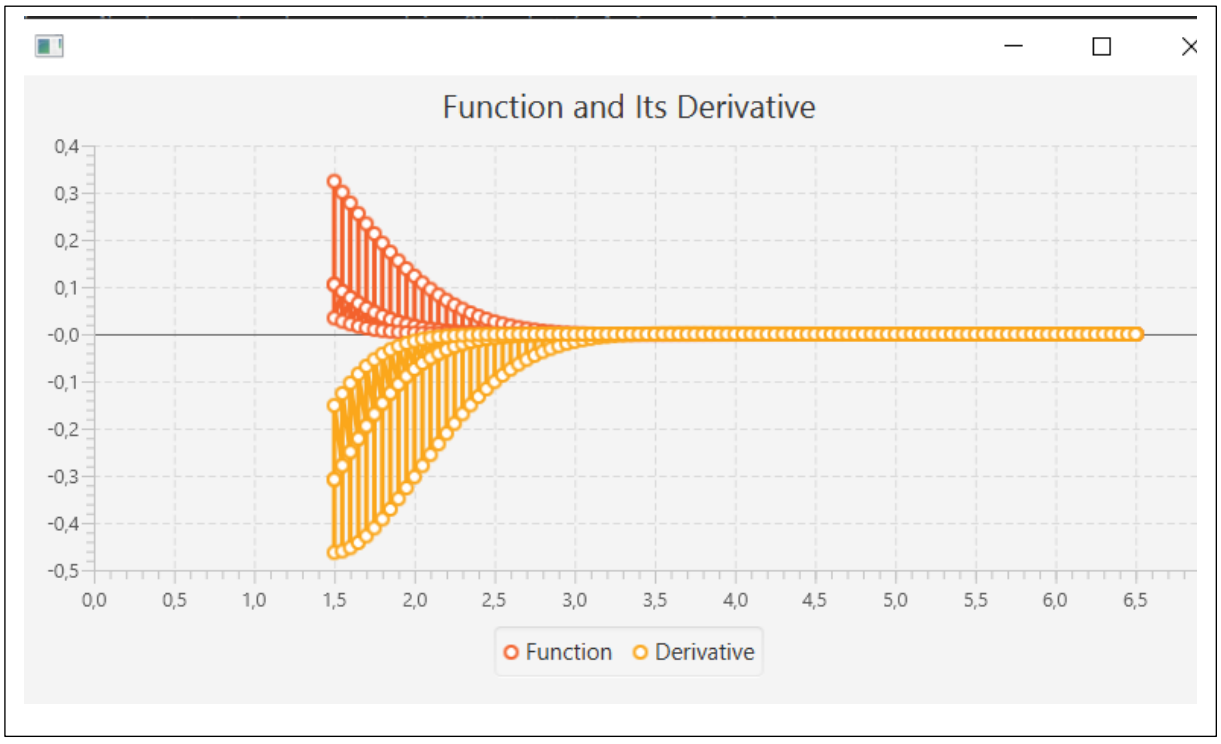
Завдання №3 Провести реінжиніринг програмного рішення (Реінжиніринг програмного забезпечення — процес створення нової функціональності або усунення помилок шляхом революційної зміни, але використовуючи вже наявне в експлуатації програмне забезпечення), додавши можливості:

- створити програму з графічним інтерфейсом користувача для відображення аналітично заданої функції та її похідної у зазначених межах (можна використати або графічну підсистему Swing або ж Java FX). Можливий зовнішній вигляд програми вказано на Рисунку.

Результати виконання завдання №3 наведено:
4. У лістингу 3 – вихідний код програми;
5. На малюнку 3.1 – результат виконання програми;

Лістинг 3. Вихідний код програми

```
NumberAxis xAxis = new NumberAxis();
NumberAxis yAxis = new NumberAxis();
LineChart<Number, Number> chart = new LineChart<>(xAxis, yAxis);
chart.setTitle("Function and Its Derivative");

XYChart.Series<Number, Number> functionSeries = new XYChart.Series<>();
functionSeries.setName("Function");

XYChart.Series<Number, Number> derivativeSeries = new XYChart.Series<>();
derivativeSeries.setName("Derivative");

// Test function 1: f(x) = exp(-x^2) * sin(x)
double precision = 0.00001;
for (double x = 1.5; x <= 6.5; x += 0.05) {
    double fx = Math.exp(-x * x) * Math.sin(x);
    double dfx = (Math.exp(-x * x) * Math.cos(x)) - (2 * x * fx);

    functionSeries.getData().add(new XYChart.Data<>(x, fx));
    derivativeSeries.getData().add(new XYChart.Data<>(x, dfx));
}

// Test function 2: f(x) = exp(-a*(x^2)) * sin(x) for a = 0.5, 1.0, 1.5
double[] valuesOfA = {0.5, 1.0, 1.5};
for (double a : valuesOfA) {
    for (double x = 1.5; x <= 6.5; x += 0.05) {
        double fx = Math.exp(-a * (x * x)) * Math.sin(x);
        double dfx = (Math.exp(-a * (x * x)) * Math.cos(x)) - (2 * a * x * fx);

        functionSeries.getData().add(new XYChart.Data<>(x, fx));
        derivativeSeries.getData().add(new XYChart.Data<>(x, dfx));
    }
}
```



Малюнок 3.1 – результат виконання програми