Харківський національний університет імені В. Н. Каразіна
Факультет комп'ютерних наук

РОЗРАХУНКОВО-ГРАФІЧНА РОБОТА №1

ДИСЦИПЛІНА: «КРОС-ПЛАТФОРМНЕ ПРОГРАМУВАННЯ»

Виконав: студентка групи КС22
Мазуренко Анжеліки
Перевірив: Споров Олександр
Євгенович

Харків
2024

ЗМІСТ

Вступ

Нашою задачою є створити програмне рішення для представлення функції, заданої різними способами. У створенній системі потрібно зробити рефакторинг, проаналізувавши її недоліки, недоліки її структури з точки зору об’єктно-орієнтованого програмування та вирішити знайдені проблеми. Крім того, прибрати класи, які нам непотрібні, додати нові, покращити структуру проекту та після цього провести реінжиніринг програмного рішення - додати новий функціонал, не зламавши старий.

Мета роботи: згадати основні відомості, вивчені в рамках курсу «Об'єктно-орієнтоване програмування»: основи об'єктно-орієнтованого підходу, особливості застосування абстрактних класів і інтерфейсів, основи роботи з колекціями, основи роботи з файлами (операції читання / запису). Познайомитися з основами чисельних методів і їх реалізацією на основі інтерфейсів.

Основна частина

Класи та їх функціонал

У програмі ми маємо 2 пакети:

1.consoleTask

2. GUI

У першому у нас зібрані всі класи функціоналу, а у другому клас відповідальний за графічний інтерфейс.

Почнемо розглядати наші класи з класу FFunction:

1.Клас FFunction

Клас FFunction імплементує інтерфейс Evaluatable та представляє математичну функцію $e(-a*x^2)\cdot\sin(x)$, де a - це параметр функції, який можна встановлювати та змінювати через методи setParamFunc та getParamFunc. Також клас містить приклад використання своїх методів у методі main для обчислення та виведення значень функції для заданих значень x та a.

```
Перевірка класу FFunction
x Початок: 0,1
x Кінець:
2
x Шаг: 0,3
Параметр a: 1.0
x: 0,1000   f: 0,0988
x: 0,4000   f: 0,3318
x: 0,7000   f: 0,3947
x: 1,0000   f: 0,3096
x: 1,3000   f: 0,1778
x: 1,6000   f: 0,0773
x: 1,9000   f: 0,0256
--------------------
x: 2
a Початок: 0,2
a Кінець: 1,7
a Шаг: 0,3
Змінна x: 2.0
a: 0,2000   f: 0,4086
a: 0,5000   f: 0,1231
a: 0,8000   f: 0,0371
a: 1,1000   f: 0,0112
a: 1,4000   f: 0,0034
```

Рисунок 1 – приклад результатів виконання класу FFunction

2.Інтерфейс Evaluatable

Інтерфейс Evaluatable надає шаблон для класів, які дають змогу обчислення функцій (чи інших виразів) для заданих аргументів. Класи, які реалізують цей інтерфейс, повинні забезпечити реалізацію методу evalf(double x).

3.Клас FileManager

Цей клас FileManager є абстрактним. Він надає загальний функціонал для роботи з файлами.

public static void readFromFile(String fileName, Interpolator listInt).

Цей статичний метод використовується для читання даних з файлу та заповнення об'єкта Interpolator точками даних, представленими у файлі. Він приймає ім'я файлу fileName і об'єкт listInt типу Interpolator, який представляє колекцію точок даних. Метод відкриває файл для читання, зчитує дані рядок за рядком, парсить значення x і y з рядків і додає точки даних об'єкт listInt. Після прочитання файлу, метод закриває його.

public static void writeToFile(String fileName, Interpolator listInt) .

Цей статичний метод використовується для запису даних з об'єкта Interpolator у файл із зазначеним ім'ям fileName. Він відкриває файл для запису, записує заголовок "x,y" у перший рядок файлу, а потім записує кожну точку даних із listInt у форматі "x;y" на окремому рядку. Після запису всіх даних метод закриває файл.

4. Клас Interpolator

Також імплементить інтерфейс Evaluatable. Він визначає загальний інтерфейс класів, реалізують різні методи інтерполяції даних. Він містить абстрактні методи роботи з даними інтерполятора, а також реалізацію методу evalf для виконання інтерполяції.

5. Клас ListInterpolation

Клас ListInterpolation є реалізацією абстрактного класу Interpolator і містить конкретну логіку інтерполяції даних, а також демонстраційний код у методі main для роботи з інтерполяцією і також використовує клас FileManager для запису та читання даних із файлів.

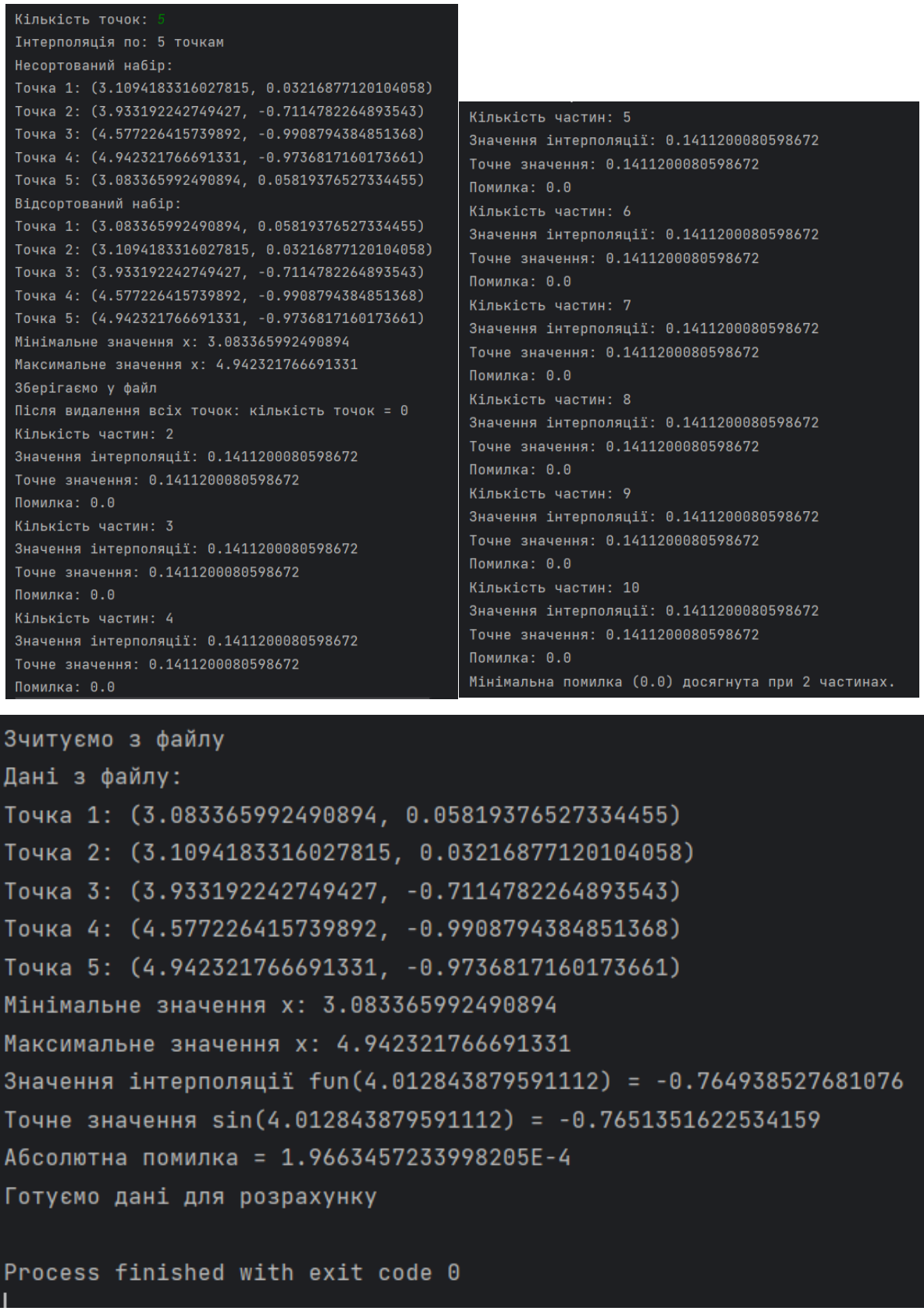


Рисунок 2 – 4 - приклад результатів виконання класу ListInterpolation

6. Клас TreeMapInterpolation

Клас TreeMapInterpolation є реалізацію інтерполяції даних на основі структури даних TreeMap для зберігання пар значень (x, y). Цей клас успадковується від абстрактного класу Interpolator, який визначає основний інтерфейс роботи з точками даних та його інтерполяцією. Записує інформацію у файл TblFunTreeMap.csv

```
Кількість точок:
5
Інтерполяція по: 5 точкам
Несортований набір:
Точка 1: (1.2158554040649179, 0.9376670218729499)
Точка 2: (1.8956032006852204, 0.947712375605437)
Точка 3: (2.9652002806265165, 0.17547907498764947)
Точка 4: (3.3559450449060115, -0.21271468520963457)
Точка 5: (4.891398002512359, -0.9840206240193561)
Відсортований набір:
Точка 1: (1.2158554040649179, 0.9376670218729499)
Точка 2: (1.8956032006852204, 0.947712375605437)
Точка 3: (2.9652002806265165, 0.17547907498764947)
Точка 4: (3.3559450449060115, -0.21271468520963457)
Точка 5: (4.891398002512359, -0.9840206240193561)
Зберігаємо у файл
Зчитуємо з файлу
Дані з файлу:
Точка 1: (1.2158554040649179, 0.9376670218729499)
Точка 2: (1.8956032006852204, 0.947712375605437)
Точка 3: (2.9652002806265165, 0.17547907498764947)
Точка 4: (3.3559450449060115, -0.21271468520963457)
Точка 5: (4.891398002512359, -0.9840206240193561)
Мінімальне значення x: 1.2158554040649179
Максимальне значення x: 4.891398002512359
Значення інтерполяції fun(3.0536267032886384) = 0.0886202279592975
Точне значення sin(3.0536267032886384) = 0.08785254730847457
Абсолютна помилка = 7.6768065082293E-4
Готуємо дані для розрахунку
```

Рисунок 5 - приклад результатів виконання класу TreeMapInterpolation

7. Клас TreeSetInterpolation

Клас TreeSetInterpolation є реалізацію інтерполяції даних на основі структури даних TreeSet. Він є спадкоємцем абстрактного класу

Interpolator і надає методи роботи з точками даних та його інтерполяції. Записує інформацію у файл TblFunTreeSet.csv

```
5
Інтерполяція по: 5 точкам
Несортований набір:
Точка 1: (1.1865412110424023, 0.9270779236140636)
Точка 2: (2.2941731248046695, 0.7495748381966085)
Точка 3: (3.345926275076539, -0.20291468955176678)
Точка 4: (4.103412797774184, -0.8202340998322145)
Точка 5: (4.104737960378005, -0.8209914095228227)
Відсортований набір:
Точка 1: (1.1865412110424023, 0.9270779236140636)
Точка 2: (2.2941731248046695, 0.7495748381966085)
Точка 3: (3.345926275076539, -0.20291468955176678)
Точка 4: (4.103412797774184, -0.8202340998322145)
Точка 5: (4.104737960378005, -0.8209914095228227)
Зберігаємо у файл
Зчитуємо з файлу
Дані з файлу:
Точка 1: (1.1865412110424023, 0.9270779236140636)
Точка 2: (2.2941731248046695, 0.7495748381966085)
Точка 3: (3.345926275076539, -0.20291468955176678)
Точка 4: (4.103412797774184, -0.8202340998322145)
Точка 5: (4.104737960378005, -0.8209914095228227)
Мінімальне значення x: 1.1865412110424023
Максимальне значення x: 4.104737960378005
Значення інтерполяції fun(2.6456395857102035) = 0.4700845377737579
Точне значення sin(2.6456395857102035) = 0.47587010531164897
Абсолютна помилка = 0.0057855675378910565
Готуємо дані для розрахунку

Process finished with exit code 0
```

Рисунок 6 - приклад результатів виконання класу TreeSetInterpolation

8. Клас PointData

Цей клас PointData представляє точку даних з координатами x і y. він представляє точку даних з координатами x та y.

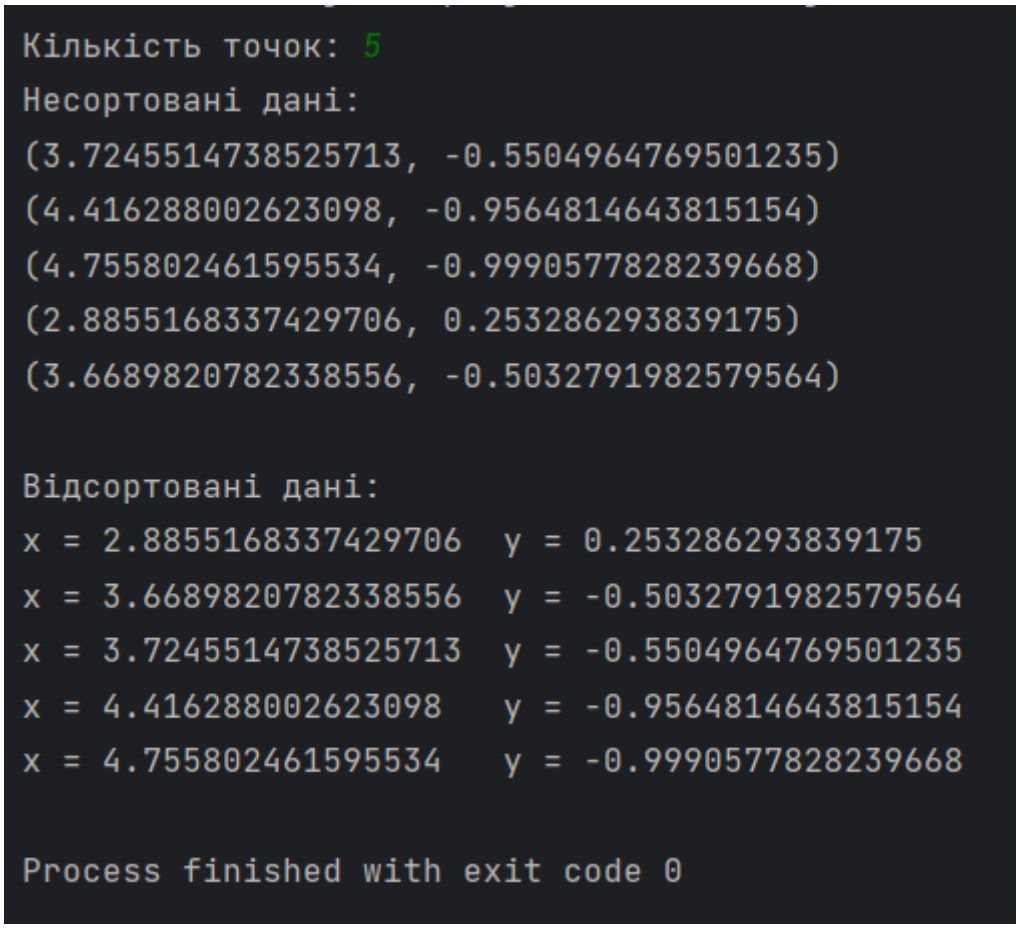


Рисунок 7 - приклад результатів виконання класу PointData

9. Клас NumMethods

Клас NumMethods містить методи для чисельного обчислення похідної функції. Клас надає зручний спосіб обчислення чисельної похідної функції із заданою точністю та порівняння її з аналітичним результатом для оцінки точності чисельного методу.

```
Кількість точок: 31
Чисельне значення sin'(0.029203673205103864) = 0.9995736030252856
Символьне значення sin'(0.029203673205103864) = 0.9995736030415051
Абсолютна помилка = 1.6219470211353837E-11

Process finished with exit code 0
```

```
Кількість точок: 97
Чисельне значення sin'(3.3292036732050976) = -0.9824526126203654
Символьне значення sin'(3.3292036732050976) = -0.9824526126243336
Абсолютна помилка = 3.9681591346152345E-12
```

Рисунок 8 – 9 - приклад результатів виконання класу NumMethods

10. Клас DerivativeApplication

У класі оголошено масив об'єктів інтерфейсу Evaluatable під назвою functs.

Масив містить чотири елементи, що представляють різні функції:

functs[0]: об'єкт класу FFunction із параметром 0.5.

functs[1]: об'єкт класу ListInterpolation (спискова інтерполяція).

functs[2]: об'єкт класу TreeMapInterpolation (інтерполяція через TreeMap).

functs[3]: об'єкт класу TreeSetInterpolation (інтерполяція через TreeSet).

Для кожної функції з масиву functs виконується обчислення значень функції та її похідної.

Викликається метод NumMethods.der для обчислення чисельної похідної кожної функції в заданому діапазоні значень x від 1.5 до 6.5 з кроком 0.5.

Результати обчислень виводяться на консоль.

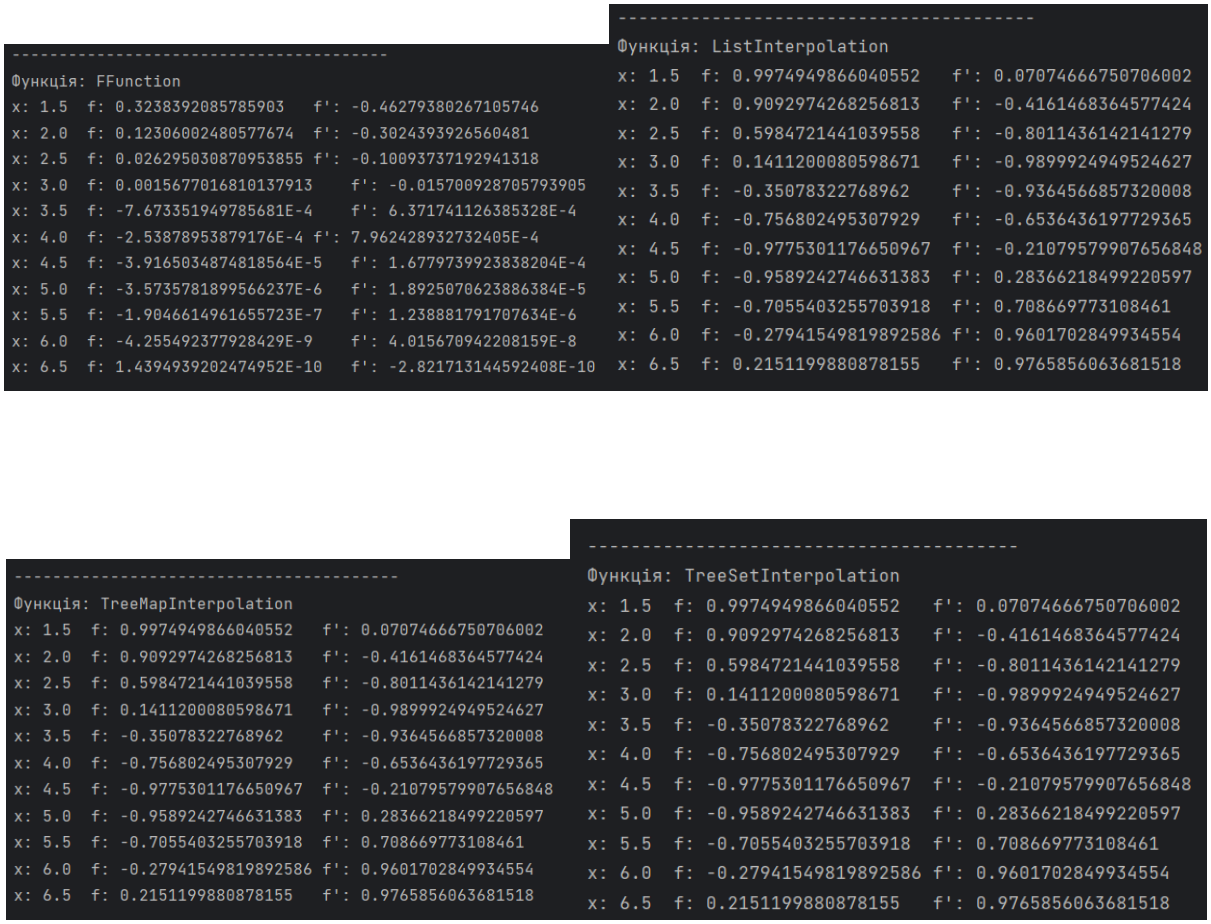


Рисунок 10 – 13 - приклад результатів виконання класу DerivativeApplication

11. Клас FunctionParam

Клас FunctionParam є програмою для роботи з математичними функціями з параметром і без параметра. Він дозволяє користувачеві вибирати тип функції, вводити дані про функції та інтервал значень, а

потім обчислювати та виводити значення функції та її похідної.

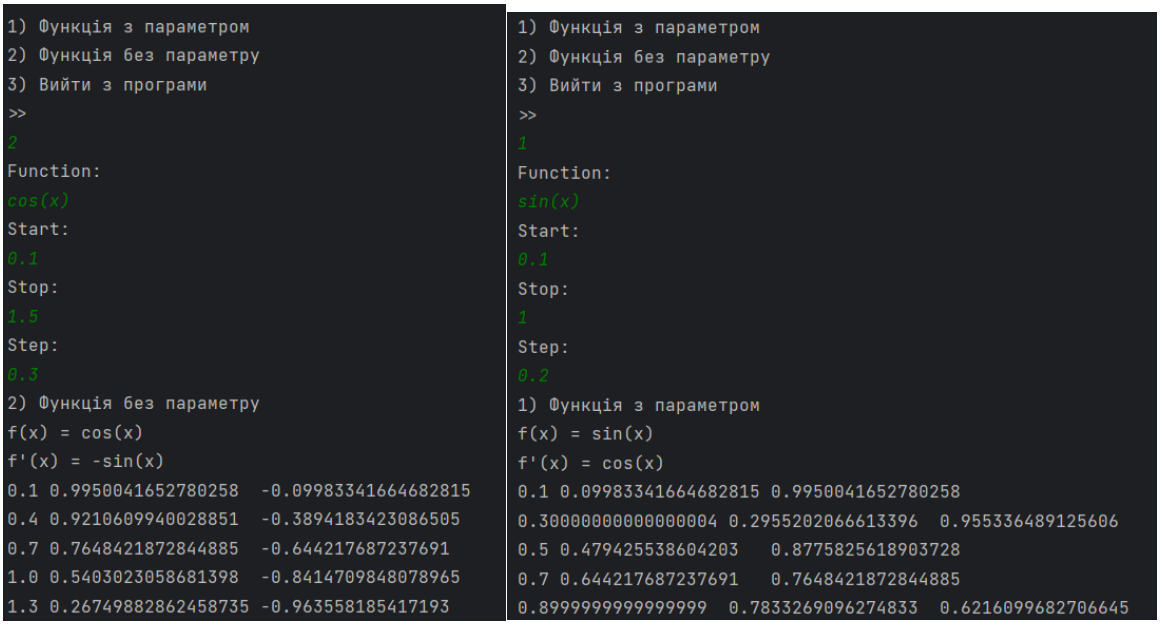


Рисунок 14 – 15 - приклад результатів виконання класу FunctionParam

Ось такі класи ми реалізували в ході роботи ,разом вони забезпечують вирішення нашої задачі. Класи взаємодіють між собою, а архітектура з використанням успадкування та інтерфейсів дозволяє розширювати та модифікувати функціональність програми, додаючи нові методи інтерполяції або змінюючи способи обчислення похідної. Такий підхід робить код більш гнучким та підтримуваним у довгостроковій перспективі.

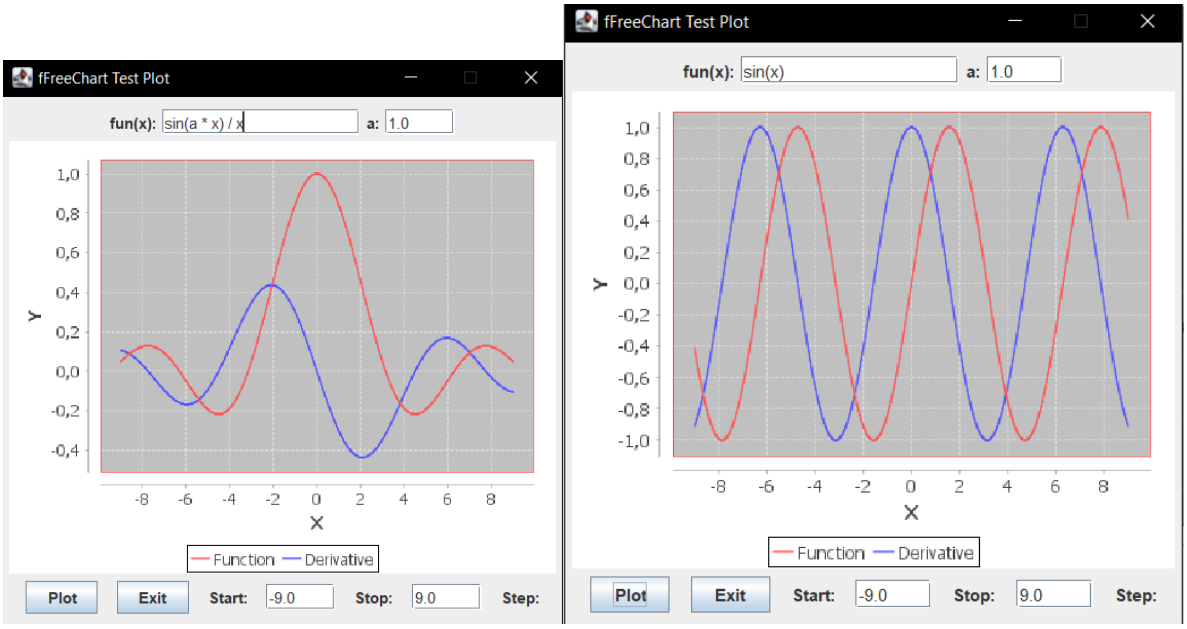
Також, у нас реалізован клас - програма графічного інтерфейсу, він знаходиться у пакеті GUI та має назву JFreeChartMainFrame

Загальний принцип роботи програми:

Користувач вводить ці функції та параметри.

Натискає кнопку "Plot", після чого відбувається побудова графіка функції та її похідної на заданому інтервалі із заданим кроком.

Ця програма корисна для візуалізації функцій та їх похідних на графіку, що допомагає у розумінні та дослідженні математичних функцій.



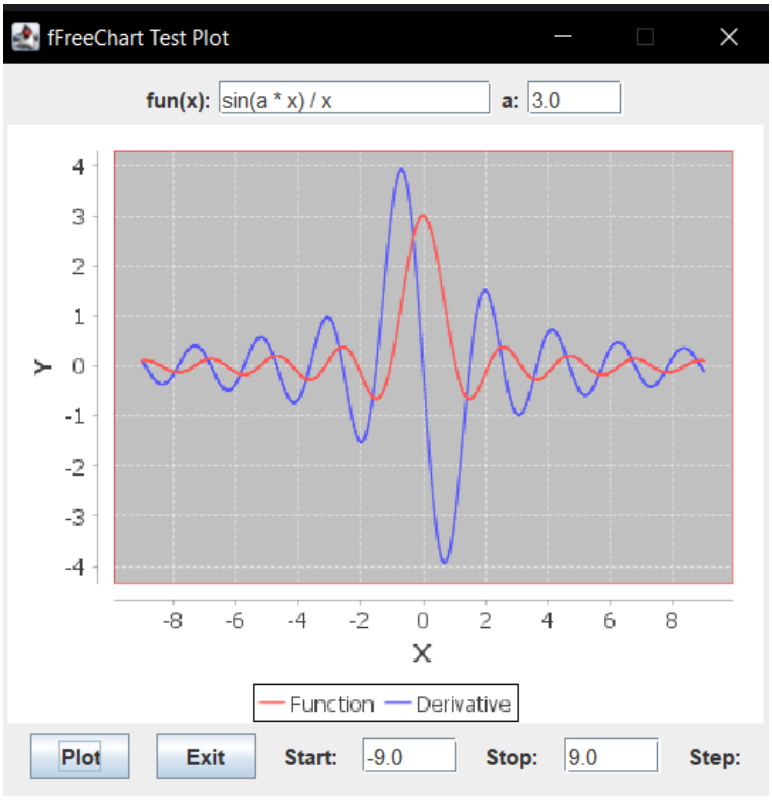


Рисунок 16 – 18 - приклад результатів виконання класу JFreeChartMainFrame

Висновок

При виконані розрахунково-графічної робити 1 ми створили 10 класів, 1 інтерфейс та 1 клас для графічного відображення. У програмі класи взаємодіють між собою та представляють 1 систему, бо під час виконання 2 завдання – рефакторингу ми прибрали класи, які нам не потрібні, та за допомогою наслідування та імплементації запобігли повторювання коду.

Додаток

Репозиторій: https://github.com/miorezu/RGR1