МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНІЙ ТЕХНІЧНИЙ УНІВЕРИТЕТ
«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»
Кафедра «Програмна інженерія та інформаційні технології управління»

КУРСОВА РОБОТА

«Розробка прикладної програми графічного інтерфейсу користувача для чисельного знаходження коренів методом хорд»

Керівник роботи:
ст. викл. каф. ПІТУ Іванов Л.В.
Виконавець:
студента гр. КН – 36а Рубан Ю.Д.

Харків – 2017

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНІЙ ТЕХНІЧНИЙ УНІВЕРИТЕТ
«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»
Кафедра «Програмна інженерія та інформаційні технології управління»

Оцінка

_____

голова комісії ас. каф. СУ

_____/ Іванов Л.В./

«___»_____2017р.

КУРСОВА РОБОТА

з курсу «Об'єктно-орієнтоване програмування»

Тема: «Розробка прикладної програми графічного інтерфейсу користувача для чисельного знаходження коренів методом хорд»

Керівник роботи:

ст. викл. каф. ПІТУ / Іванов Л.В./

« »_____2017р.

Виконавець:

студент гр. КН-36а /Рубан Ю.Д./

« »_____2017р.

Харків – 2017

ВІДГУК

на курсову роботу

студента групи КН-36а Рубана Ю. Д.

«Розробка прикладної програми графічного інтерфейсу користувача для чисельного знаходження кореня рівняння методом хорд»

Курсова робота присвячена розробці прикладної програми графічного інтерфейсу користувача для чисельного знаходження коренів рівняння методом хорд.

Під час виконання була спроектована та реалізована програма графічного інтерфейсу користувача для чисельного знаходження коренів рівняння

Робота виконана повністю і може бути захищена.

Керівник роботи:

ст. викл. каф. ПШТУ Іванов Л.В.

РЕФЕРАТ

КР: 24 с., 12 рис., 9 джерел

Ключові слова: C#, Windows Forms, МЕТОД ХОРД, МЕТОД ІНТЕРПОЛЯЦІЇ ЛАГРАНЖА, ПОШУК ТОЧКИ ПЕРЕТИНУ ДВОХ ГРАФІКІВ

Метою даної курсової роботи було створення програмного застосунку для чисельного знаходження точки перетину двох графіків.

Курсова робота виконується з метою закріплень знань та вдосконалення навичок, отриманих з курсу "Об'єктно-орієнтоване програмування", а саме знань про наслідування та використання графічних застосунків для інтерфейсу користувача.

В ході виконання курсової роботи було розроблене програмне забезпечення, відповідно варіанту завдання, використовуючи мову C#. Під варіантом завдання слід розуміти знаходження коренів рівняння методом хорд, де рівняння задані поліномом та точками, для яких використовується метод інтерполяції Лагранжа для знаходження значення функції в точці. Розроблене програмне забезпечення дозволяє знаходити корені рівняння, будувати графік отриманої функції та взаємодіяти з ним, зберігати та завантажувати дані з XML файлу, а також зберігати отримані дані у вигляді звіту в форматі html-сторінки.

РЕФЕРАТ

КР: 24 с., 12 рис., 9 источников

Ключевые слова: C#, Windows Forms, метод хорд, метод интерполяции Лагранжа, ПОИСК точки пересечения двух графиков

Целью данной курсовой работы было создание программного приложения для численного нахождения точки пересечения двух графиков.

Курсовая работа выполняется с целью креплений знаний и совершенствование навыков, полученных по курсу "Объектно-ориентированное программирование", а именно знаний о подражании и использования графических приложений для интерфейса.

В ходе выполнения курсовой работы было разработано программное обеспечение, согласно варианту задания, используя язык C#. Под вариантом задания следует понимать нахождения корней уравнения методом хорд, где уравнения заданные полиномом и точками, для которых используется метод интерполяции Лагранжа для нахождения значения функции в точке. Разработанное программное обеспечение позволяет находить корни уравнения, строить график полученной функции и взаимодействовать с ним, хранить и загружать данные из XML файла, а также сохранять полученные данные в виде отчета в формате html-страницы.

ABSTRACT

CW: 24 p., 12 img., 9 sources

Keywords: C#, Windows Forms, Chord Method, Lagrange interpolation method, POINT SEARCH OF TWO GRAPHICS REVIEW

The purpose of this course work was to create a software application for numerically finding the point of intersection of the two graphs.

Coursework is carried out in order to consolidate knowledge and improve the skills derived from the course "Object-Oriented Programming", namely knowledge of the imitation and use of graphical applications for the user interface.

During the course work, software was developed, according to the task variant, using the C# language. Under the variant of the task it is necessary to understand the finding of the roots of the equation by the method of dichotomy, where the equations are given by the polynomial and the points for which the method of Lagrange interpolation is used for finding the value of the function at the point. The developed software allows you to find the roots of the equation, build a graph of the received function and interact with it, store and load the data from the XML file, and also store the received data in the form of a report in html-page format.

ЗМІСТ

ВСТУП

Розробка прикладної програми графічного інтерфейсу користувача для чисельного знаходження кореня рівняння методом хорд є актуальною і важливою темою. Пошук наближеного кореня рівняння відіграє ключову роль у математиці, і його пошук є пріоритетними задачами в усіх точних науках. Метою роботи є створення програмного продукту, використання якого дозволить вирішити задачу знаходження коренів рівняння

$$f(x)=g(x).$$

На сьогоднішній день тема наближеного розв’язку нелінійних рівнянь є досить актуальною у багатьох технічних спеціальностях. Пошук наближених коренів рівняння відіграє ключову роль у математиці та є однією з найприорітетніших задач у більшості точних наук.

Під час багатьох інженерних розрахунків є необхідним обчислення коренів рівнянь з певною точністю. Зазвичай, розв’язки нелінійних рівнянь знаходять наближено, на основі відомих даних. Проте, використання даного способу приведе до результату з певною похибкою.

За допомогою даного програмного забезпечення користувачеві надається можливість знаходження коренів нелінійного рівняння методом хорд на проміжку з певною точністю, а також переглядати готовий розв’язок у графічному вигляді.

1 МАТЕМАТИЧНІ МОДЕЛІ ТА АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1.1 Метод хорд.

Детальна інформація про метод хорд [1].

Нехай задано нелінійне рівняння $f(x) = 0$, де $f(x)$ на відрізку $[a; b]$ має неперервні похідні першого і другого порядків, які зберігають сталі знаки на цьому відрізку.

Ідея методу хорд полягає в тому, що на достатньо малому відрізку $[a; b]$ дугу кривої замінюють хордою і за наближений розв'язок береть точку перетину даної хорди з віссю абсцис. Розглянемо випадок (рис 1.1), коли перша і друга похідні мають однакові знаки, тобто $f'(x) \cdot f''(x) > 0$:

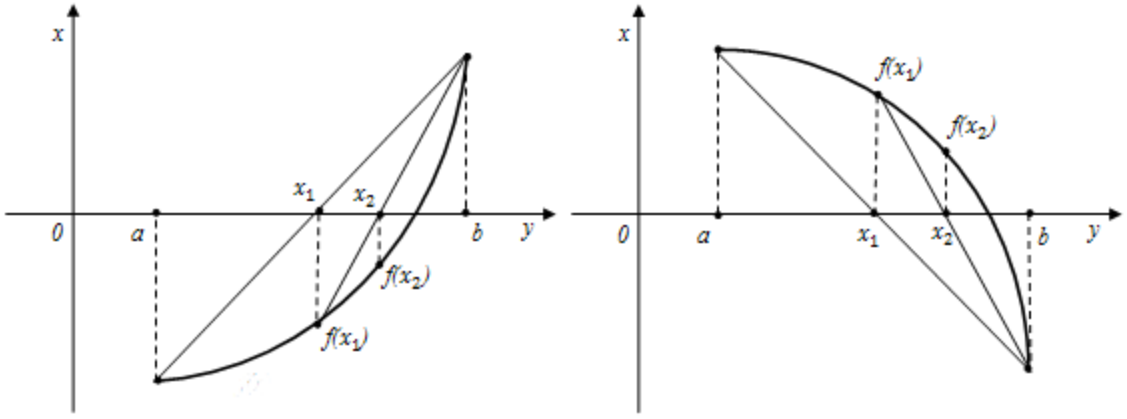

Рисунок 1.1 - Випадок методу хорд, коли добуток першої і другої похідної більшої за нуль.

Рівняння хорди — це рівняння прямої, що проходить через точки $A(a, f(a))$ і $B(b, f(b))$. А загадьний вигляд рівняння прямої, яка проходить через дві точки (x_1, y_1) і (x_2, y_2) має вигляд:

$$\frac{x - x_1}{x_2 - x_1} = \frac{y - y_1}{y_2 - y_1}$$

Тобто, для того, щоб знайти рівняння хорди, підставимо замість точок (x_1, y_1) *і* (x_2, y_2) точки *A* та *B*. В результаті отримаємо:

$$\frac{y-f(a)}{f(b)-f(a)}=\frac{x-a}{x-b}$$

Нехай x_1 — точка перетину хорди з віссю *OX*, так як $y=0$, то:

$$x_1=a-\frac{(b-a)f(a)}{f(b)-f(a)}$$

де x_1 — наближене значеня шуканого кореня. Опустимо перпендикуляр із цієї точки до графіка функції, утвориться якась точка *A1*. Проводимо хорду через точки *A1B* і аналогічним чином знаходимо наступне наближене значеня кореня:

$$x_2=x_1-\frac{(b-x_1)f(x_1)}{f(b)-f(x_1)}$$

Тобо, у загальному випадку формула методу хорд матиме вигляд:

$$x_{i+1}=x_i-\frac{(b-x_i)f(x_i)}{f(b)-f(x_i)}$$

Якщо перша і друга похідні мають різні знаки, тобто $f'(x)\cdot f''(x)<0$, то всі наближення до кореня відшукуються з боку правої межі відрізка (рис 1.2):

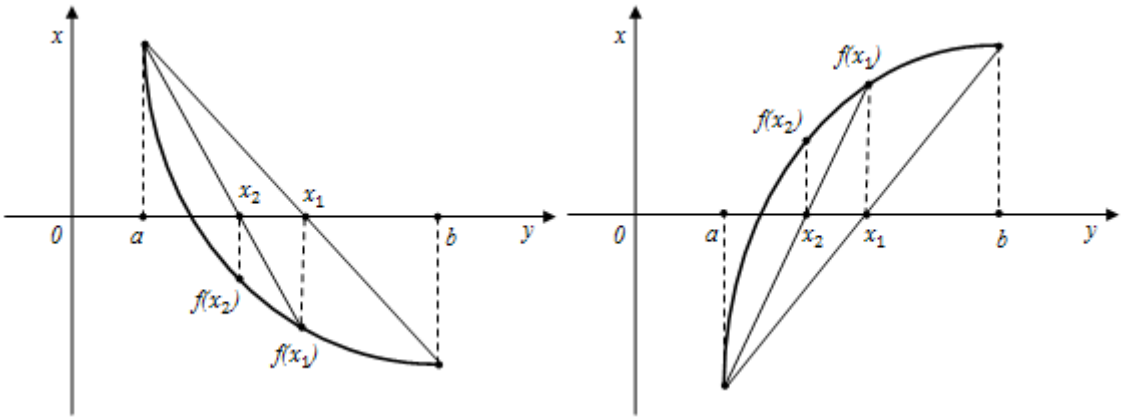

Рисунок 1.2 - Випадок методу хорд, коли добуток першої і другої похідної менший нуля

Обчислюються за наступною формулою:

$$x_{i+1} = x_i - \frac{(x_i - a)f(x_i)}{f(x_i) - f(a)}$$

Ітераційний процес методу хорд продовжеється до тих пір, поки не буде виконуватись умова: $|x_{i+1} - x_1| < \varepsilon$, тобто коли модуль різниці між двома сусідніми значеннями наближень отриманих за методом хорд стане меншим за ε (де ε - задана похибка обчислень).

1.2 Метод інтерполяції Лагранжа

Детальна інформація про метод інтеполяції [2]

Нехай функція f(x) задана набором точок (x_i; y_i) на проміжку $[a,b]$:

$$y_i = f(x_i),\ i = 0,1,\ldots,n,\ a \le x_i \le b$$

Задача інтерполяції – знайти функцію F(x), яка приймає в точках x_i ті ж самі значення y_i. Тоді, умова інтерполяції:

$$F(x_i) = y_i$$

При цьому передбачається, що серед значень x_i нема одинаковых. Точки x_i називають вузлами інтерполяції.

Якщо F(x) шукається тільки на проміжку [a, b] – то це задача інтерполяції, а якщо за границями проміжка, то це задача екстраполяції.

Задача знахожденя інтерполяційної функції F(x) має багато розв’язків, так як через задані точки x_i, y_i можна провести нескінченно багато кривих, кожна з

яких буде графіком функції, для якої виконані всі умови інтерполяції. Для практики важливий випадок інтерполяції функції многочленами:

$$F(x) = P_m(x_i) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + \ldots + a_m \cdot x^m \; i = 0,1,\ldots,m$$

При цьому шуканий поліном називається інтерполяційним поліномом.

При побудові одного многочлена для всього розглянутого інтервалу [a, b] для знаходження коефіцієнтів многочлена необхідно вирішити систему рівнянь, побудовану на основі полінома. Дана система містить n+1 рівняння, отже, з її допомогою можна визначити n+1 коефіцієнт. Тому максимальний степінь інтерполяційного многочлена m=n, і многочлен приймає вид

$$P_n(x_i) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + \ldots + a_n \cdot x^n, \; i = 0,1,\ldots,n$$

При глобальної інтерполяції на всьому інтервалі [a, b] будується єдиний многочлен. Однією з форм запису інтерполяційного многочлена для глобальної інтерполяції є многочлен Лагранжа:

$$L_n(x) = \sum_{i=0}^{n} y_i \cdot l_i(x)$$

де $l_i(x)$ – базисні многочлени степені *n*:

$$l_i(x) = \prod_{\substack{j=1 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j} = \frac{(x - x_0)(x - x_1)\ldots(x - x_{i-1})(x - x_{i+1})\ldots(x - x_n)}{(x_i - x_0)(x_i - x_1)\ldots(x_i - x_{i-1})(x_i - x_{i+1})\ldots(x_i - x_n)}$$

Тобто многочлен Лагранжа можна записати у вигляді:

$$L_n(x) = \sum_{i=0}^{n} y_i \cdot \prod_{\substack{j=1 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}$$

Многочлен $l_i(x)$ задовольняє умові $l_i(x_j) = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$. Ця умова означає, що многочлен дорівнює нулю при кожному x_j окрім x_i, то $x_0, x_1, \ldots x_{i-1}, x_{i+1}, \ldots x_n$ – корені цього многочлена. Таким чином, ступінь многочлена $L_n(x)$ дорівнює *n* і при $x \neq x_i$ звертаються в нуль всі складові суми, крім доданка з номером i=j, рівного y_i.

Похибка інтерполяції методом Лагранжа залежить від властивостей функції f(x), від розташування вузлів інтерполяції і точки x. Поліном Лагранжа має малу похибку при невеликих значеннях *n* (*n*<20). При великих n похибка починає рости, що свідчить про те, що метод Лагранжа не сходиться (тобто його похибка не зменшується з ростом *n*).

Многочлен Лагранжа в явному вигляді містить значення функцій у вузлах інтерполяції, тому він зручний, коли значення функцій змінюються, а вузли інтерполяції незмінні. Число арифметичних операції, необхідних для побудови многочлена Лагранжа, пропорційно n^2 і є найменшим для всіх форм записи. До недоліків цієї форми запису можна віднести те, що зі зміною числа вузлів доводиться все обчислення проводити заново.

Кусково-лінійна і кусочно-квадратична локальні інтерполяції є окремими випадками інтерполяції многочленом Лагранжа.

1.3 Вимоги уніфікованої мови моделювання

Використання мови UML ґрунтується на наступних загальних принципах моделювання:

- абстрагування – у модель слід включати тільки ті елементи проектної системи, які мають безпосереднє відношення до виконання своїх функцій або свого цільового призначення. Інші елементи опускаються, щоб не ускладнювати процес аналізу та дослідження моделі;
- багатомодельність – ніяка єдина модель не може з достатнім ступенем точності описати різні аспекти системи. Допускається описувати систему деяким числом взаємозалежних уявлень, кожне з яких відображає певний аспект її поведінки або структури;
- ієрархічна побудова – При описі системи використовуються різні рівні абстрагування і деталізації в рамках фіксованих уявлень. При цьому перше представлення системи описує її в найбільш загальних рисах і є поданням концептуального рівня, а наступні рівні розкривають різні аспекти системи з зростаючої ступенем деталізації аж до фізичного рівня. Модель фізичного рівня в UML відображає компонентний склад проектованої системи з точки зору її реалізації на апаратурної і програмної платформах конкретних виробників.

Для точного опису системи в UML використовуються, так звані, загальні механізми :

- специфікації (specifications);
- доповнення (adornments);
- ділення (common divisions);
- розширення (extensibility mechanisms).

Види діаграм UML

Графічні зображення моделей системи в UML називаються діаграмами . У термінах мови UML визначені наступні їх види:

- діаграма варіантів використання або прецедентів (use case diagram)
- діаграма класів (class diagram)
- діаграми поведінки (behavior diagrams)
- діаграма станів (statechart diagram)
- діаграма діяльності (activity diagram)

- діаграми взаємодії (interaction diagrams)
- діаграма послідовності (sequence diagram)
- діаграма кооперації (collaboration diagram)
- діаграми реалізації (implementation diagrams)
- діаграма компонентів (component diagram)
- даграмма розгортання (deployment diagram)

Кожна з цих діаграм конкретизує різні уявлення про моделі системи. При цьому, діаграма варіантів використання представляє концептуальну модель системи, яка є вихідною для побудови всіх інших діаграм. Діаграма класів є логічною моделлю, що відбиває статичні аспекти структурної побудови системи, а діаграми поведінки, які також є різновидами логічної моделі, відображають динамічні аспекти її функціонування. Діаграми реалізації слугують для подання компонентів системи і відносяться до її фізичної моделі.

1.4 Методи розробки програми

Для розробки програми, що відповідає заданому варіанту, використовувалась об’єктно орієнтована мова програмування C# та інтегрована середа розробки Visual Studio.

Для розробки графічного інтерфейсу використовувався фреймоворк .NET, побудований на мові програмування C#, який надає можливість використовувати погоджені об’єктно орієнтовані засоби.

Для моделювання графічного інтерфейсу використовувались Windows Forms, що дозволяє швидко та зручно розробити графічний інтерфейс програми.

1.5 Постановка задачі

Необхідно розробити програмне забезпечення яке реалізує створення прикладної програми графічного інтерфейсу користувача для чисельного знаходження мінімуму між двома функціями.

Користувач повинен мати можливість побачити результати вирішення рівняння, побачити графік, побудований з даних наборів крапок. Також повинна бути функція збереження та зчитування даних з XML.

Описані вище завдання повинні бути реалізовані в ході розробки програми для виконання завдання курсової роботи. Програма повинна мати привітний до користувача дизайн.

В результаті виконання курсової роботи повинна бути розроблена програма, яка буде працювати без помилок, побудована за допомогою обраних програмних засобів.

В ході роботи повинна бути обрана платформа (операційна система) для реалізації програмного продукту.

Для розроблення програмного продукту потрібно обрати програмні засоби, за допомогою яких будуть реалізовані всі завдання, поставлені до програми.

Виділимо основні етапи розробки проектування застосунку:

- реалізація програмного коду програми;
- тестування програмного забезпечення. Найбільший інтерес для проблематики розглянутого питання представляють наступні етапи розробки:

Величезне значення має те, яку парадигму програмування (парадигму програмування також необхідно розглядати як засіб розробки) необхідно використовувати при написанні програми. Як приклад основних парадигм необхідно привести наступне:

- функціональне програмування;
- логічне програмування;
- об'єктно-орієнтоване програмування

На етапі реалізації програмного коду виконується кодування окремих компонент програми. Тим не менше, серед засобів розробки програмного коду необхідно виділити наступні основні види засобів:

- методи та методики алгоритмування;
- мови програмування (C++, C, Java, C#, php і багато інших);
- засоби створення користувальницького інтерфейсу (JavaFX, WPF, QT, GTK+ і т.д.);
- засоби отримання виконуваного коду (MS Visual Studio, IntellijIDEA і багато інших);
- відладчики (MS Visual Studio, Intellij IDEA і т.д.).

Після вибору операційної системи та програмних засобів необхідно розробити відповідне програмне забезпечення, налагодити його та протестувати.

1 ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Вимоги до програмного забезпечення

Діаграма варіантів використання є вихідним концептуальним уявленням або концептуальною моделлю системи в процесі її проектування і розробки.

Розробка діаграми варіантів використання програмного забезпечення дозволить:

- визначити загальні межі і контекст модельованої предметної області на початкових етапах проектування системи;
- сформулювати загальні вимоги до функціональної поведінки програмного забезпечення;
- розробити вихідну концептуальну модель програмного забезепечення для його подальшої деталізації у формі логічних і фізичних моделей;
- підготувати вихідну документацію для взаємодії розробників програмного забезпечення з його замовниками і користувачами.

Суть даної діаграми полягає в наступному: проектована система представляється у вигляді безлічі сутностей або акторів, що взаємодіють з системою за допомогою так званих варіантів використання та містять так звані зв’язки між собо. Ці зв’язки відображають взаємодію (такі, як включення або розширення, або include та extend) або переход від одного варіанту використання, до іншого.

Вимоги до програмного забезпечення відображаються на діаграмі варіантів використання, представленої на рисунку 2.1.

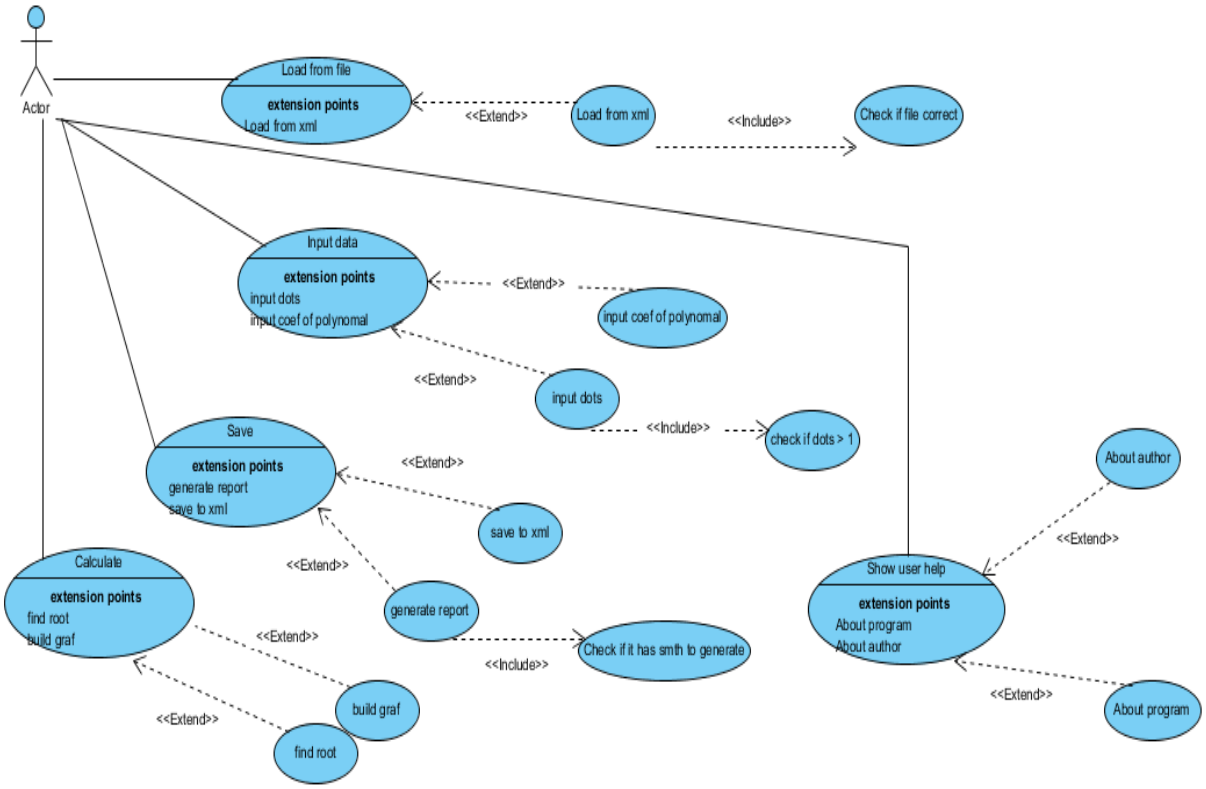

Рисунок 2.1 – Діаграма варіантів використання

Основні варіанти використання (use cases) – це взаємодія користувача з меню допомоги, файл з графіком та використання основного функціонала – введення функцій та діапазону обчислення, а також отримання звіту (у вигляді html сторінки) та рішення рівняння у вигляді точок перетину.

При взємодії з пунктом меню допомоги користувач має можливість переглянути автора програми, а також отримати постановку задачі, яка розв’язується програмою.

2.2 Діаграми послідовності для різних варіантів використання

Діаграми взаємодії описують послідовність повідомлень, якими обмінюються між собою об'єкти. Завдяки діаграмам взаємодії описується поведінка системи під час реалізації різних варіантів використання.

Діаграма послідовності показана на рисунку 2.2.

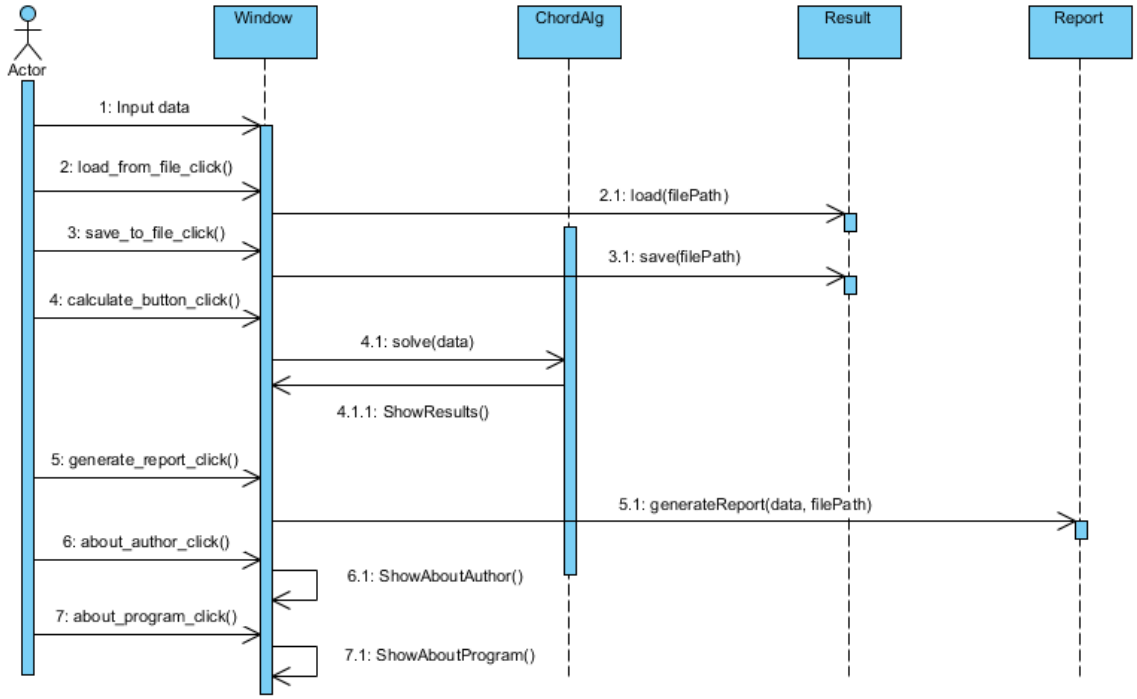

Рисунок 2.2 – Діаграма послідовності

2.3 Діаграма класів

Діаграма класів – статичне представлення структури моделі. Відображає статичні (декларативні) елементи, такі як: класи, типи даних, їх зміст та відношення.

На діаграмі класів (рисунок 2.3) програми основними класами є класи Function, Dot, Lagr, Polynomal, Result, Report, ChordAlg.

Клас Fucntion представляє собою інтерфейс, похідні класи якого реалізують функцію f(x)

Класи Polynomal і Lagr є похідними від Function, і реалізують функціонал полінома і інтерполяціонної функції відповідно.

Клас Dot необхідний для представлення точок (x; y), які використовує клас Lagr.

Клас Result використовується для зберігання та завантаження даних з XML файлів.

Клас Report є похідним від Result і реалізує функціонал генерації HTML звіту

Клас ChordAlg є статичним класом, який реалізує функцію пошуку коренів рівняння f(x)=g(x)

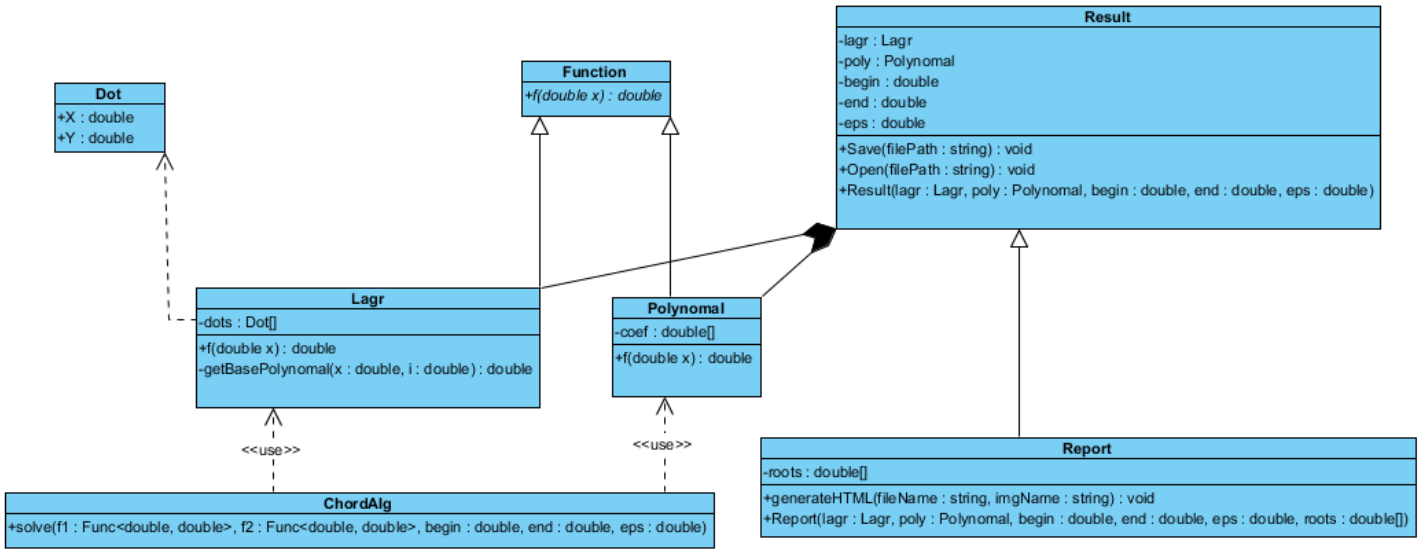

Рисунок 2.3 – Діаграма класів

2.4 Діаграми пакетів

Діаграми пакетів уніфікованої мови моделювання(UML) відображають залежності між пакетами, з яких і складається модель (рисунок 2.4)

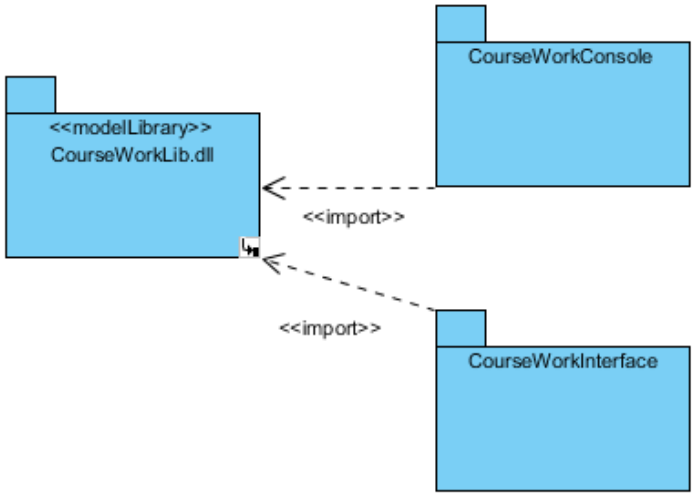

Рисунок 2.4 – Діаграма пакетів

2 ОПИС ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Рекомендації з використання розробленого програмного забезпечення

При запуску програмного забезпечення відкривається робоче вікно, яке представлене на рисунку 3.1.

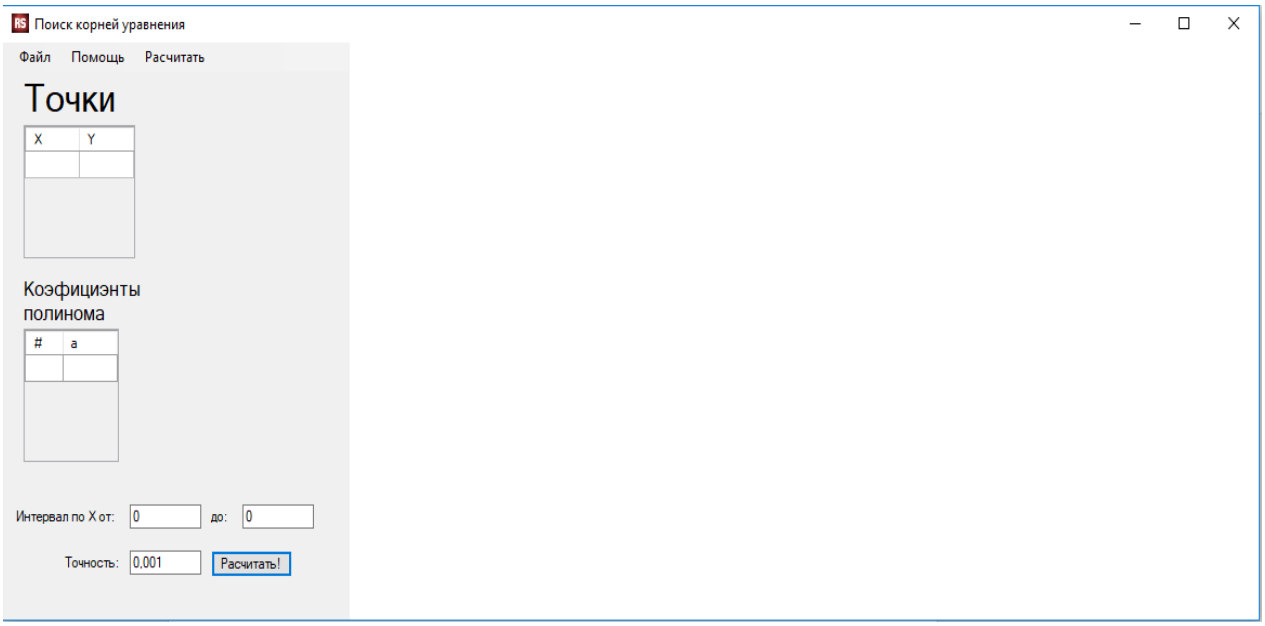

Рисунок 3.1 – Головне вікно програми

Основні елементи інтерфейсу: меню, таблиця для введення точок, таблиця для введення коефіцієнтів поліному, кнопка обчислення коренів і малювання графіка, поля для введення діапазону розрахунку коренів.

Меню містить такі пункти: Файл, Помощь, Расчитать.

Пункт Файл дозволяє зберегти або відкрити xml файл, а також сгенерувати HTML звіт. Пункт Расчитать дозволяє обчислити коріння рівняння f(x) – g(x) = 0, а також малювати графік функції. Пункт Помощь дозволяє отримати інформацію щодо автора програми та постанову задачі, яку вирішує програма.

На рисунку 3.2 зображено приклад вікна програми з введеними даними та результатом.

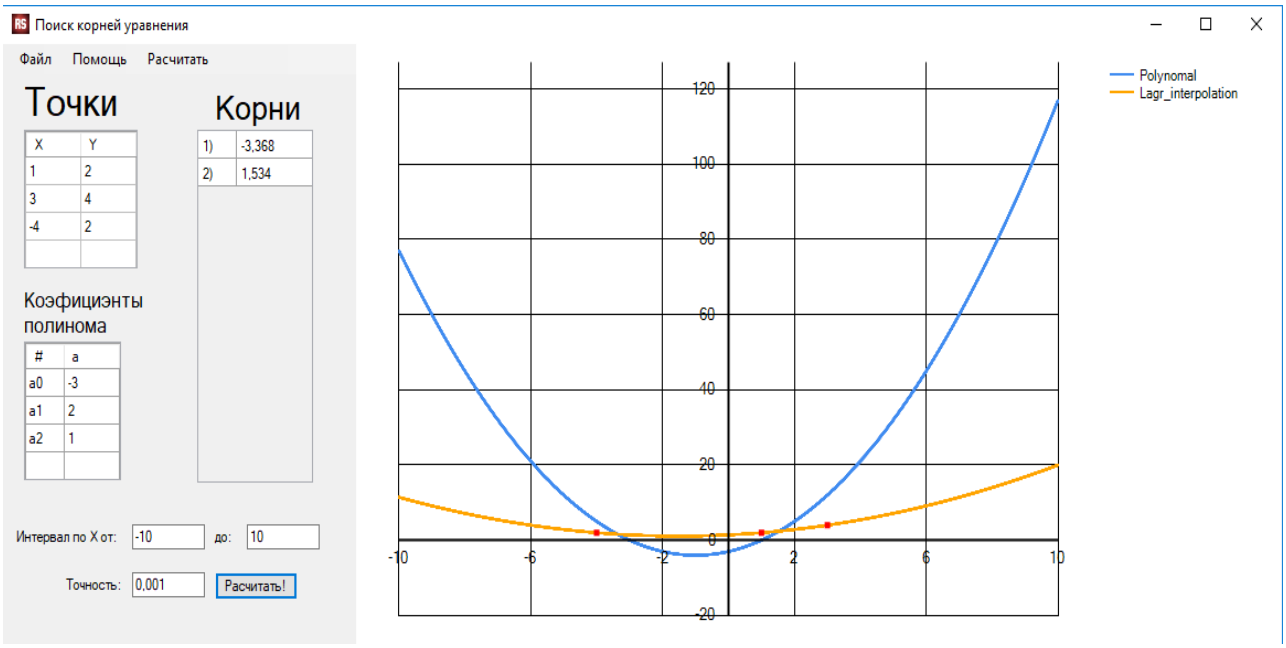

Рисунок 3.2 – Заповнення даних та відображення результату

3.2 Аналіз результатів

Після знаходження коренів рівняння $f(x)=g(x)$ користувач може переглянути звіт роботи програми. Звіт містить вхідні дані, такі, як набір точок, такі, як набір коренів та графік функцій. Приклад вікна із звітом зображено на рисунку 3.3.

Звіт

У результаті розв'язання рівняння y(x)=g(x) з такими вихідними даними:

Дані для функції *y(x)*

#	a
0	-3
1	2
2	1

Дані для функції *g(x)*

x	y
1	2
3	4
-4	2

Значення параметру *eps*: 0,001

Значення параметру *begin*: -10

Значення параметру *end*: 10

1 Корінь: -3,368

2 Корінь: 1,534

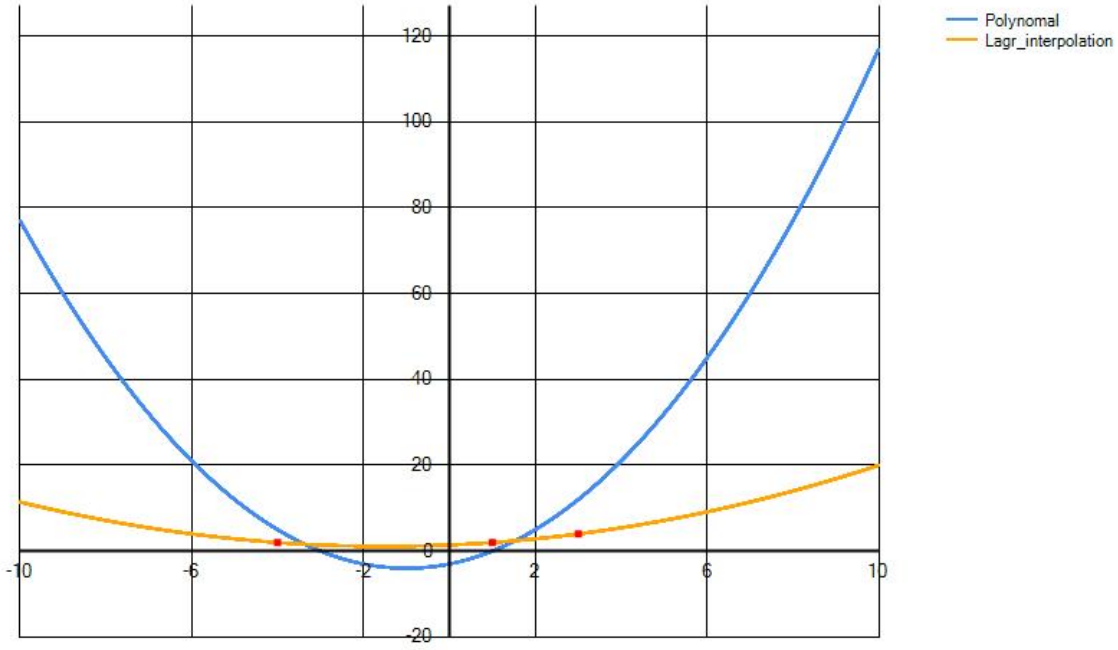

Рисунок 3.3 – Приклад звіту

3.3 Тестування програмного продукту

Тестування програмного забезпечення – це процес технічного дослідження, призначений для виявлення інформації про якість продукту відносно контексту, в якому він має використовуватись. Тестування – це одна з технік контролю якості, що включає в себе:

- планування робіт (Test Management);
- проектування тестів (Test Design);
- виконання тестування (Test Execution);
- аналіз отриманих результатів (Test Analysis).

Види тестування програмного забезпечення

За ступенем автоматизації:

- ручне тестування (manual testing);
- автоматизоване тестування (automated testing);
- напівавтоматизоване тестування (semiautomated testing);

За ступенем підготовленості до тестування:

- тестування по документації (formal testing);
- тестування ad hoc або інтуїтивне тестування (ad hoc testing) — тестування без тест плану та документації, що базується на методиці передбачення помилки та власному досвіді тестувальника.

За знанням системи:

- тестування чорного ящика (black box);
- тестування білого ящика (white box);
- тестування сірого ящика (grey box);

За ступенем ізольованості компонентів: - компонентне (модульне) тестування (component/unit testing); - інтеграційне тестування (integration testing); - системне тестування (system/end-to-end testing);

За часом проведення тестування:

1) альфа-тестування (alpha testing): - тестування при прийманні або Димове тестування (smoke testing);

- тестування нової функціональності (new feature testing);

- регресивне тестування (regression testing); - тестування при здачі (acceptance testing).

2) бета-тестування (beta testing).

Тестування програмного продукту проходило в процесі розробки. При різних вхідних даних спостерігалася реакція на них. При виявлені помилки вона виправлялася. Також проводилося тестування релізу.

У разі спроби використати програму з хибними вхідними даними, які були завантажені з XML файлу данні не будуть завантажені.

3.4 Результати тестування програми

Розроблений програмний продукт тестувався введенням різних вхідних даних та реагував по різному. Результати тестування показані на рисунках 3.4 – 3.6.

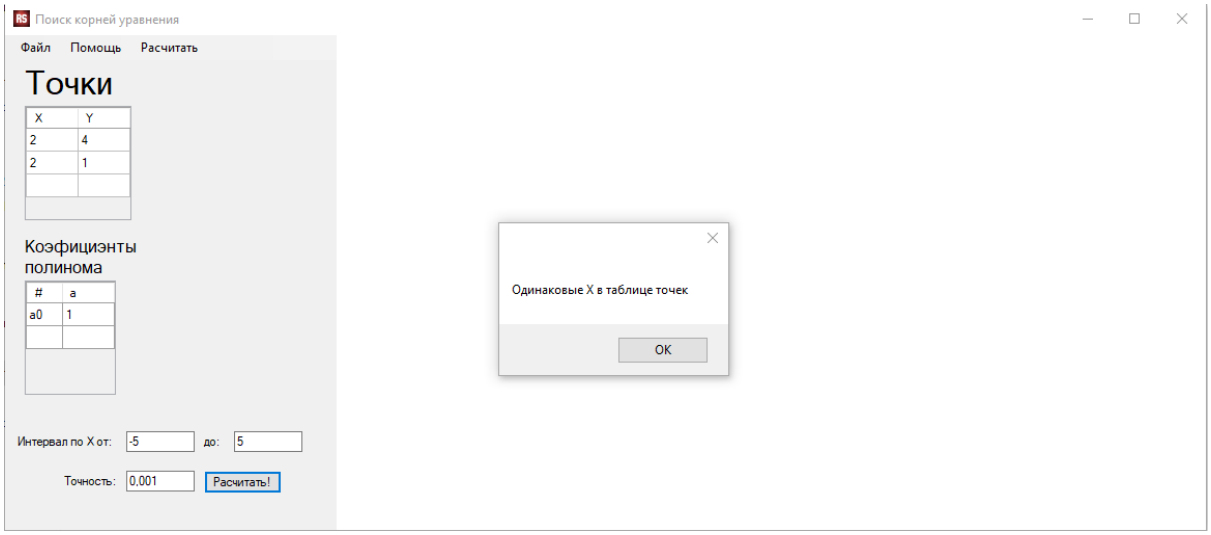

Рисунок 3.4 – Випадок, коли точки співпадають

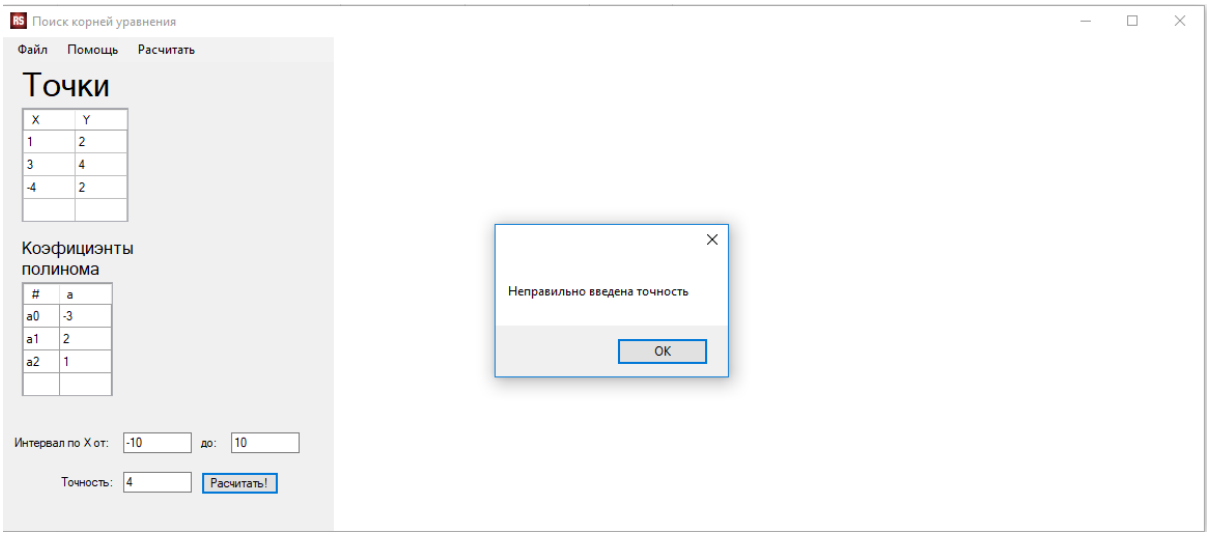

Рисунок 3.5 – Випадок, коли неправильно введена точність

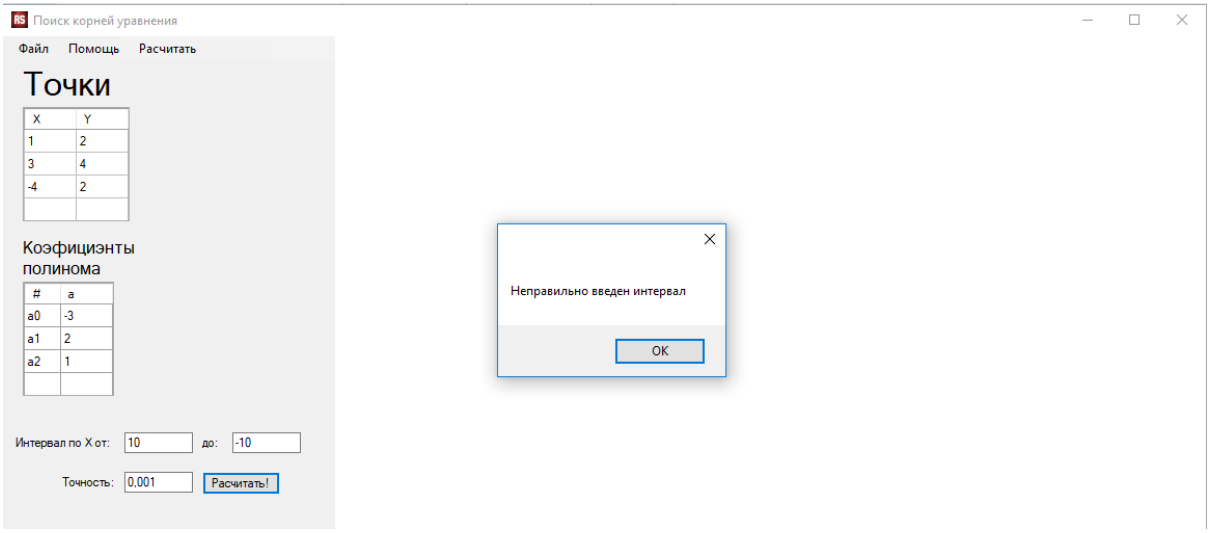

Рисунок 3.6 – Випадок, коли неправильно введений інтервал пошуку

ВИСНОВКИ

Під час виконання роботи було розроблене програмне забезпечення, яке будує функцію з набору точок методом лінійної інтерполяції та знаходить корені рівняння методом хорд.

Програма графічного інтерфейсу, яка надана у роботі, має можливість знаходження корені рівняння функції $f(x)=g(x)$ методом хорд. Було розроблено ряд UML діаграм та створений графічний інтерфейс для реалізації програмного забезпечення.

Можливості програмного продукту:

1) зчитування даних з XML-документу;

2) запис даних до XML-документу;

3) пошук коренів;

4) генерація звіту;

5) перегляд звіту;

6) отримання підказок.

У ході виконання роботи були отримані навички створення програмного забезпечення засобами Windows Forms. Були вивчені та досліджені метод лінійної інтерполяції та процесс знаходження коренів рівняння методом хорд.

СПИСОК ДЖЕРЕЛ ІНФОРМАЦІЇ

1 Метод хорд// http://www.mathros.net.ua/znahodzhennja-nablyzhenogo-rozvjazku-nelinijnogo-algebraichnogo-rivnjannja-metodom-hord.html, 16.11.2017

2 Многочлен Лагранжа// http://aco.ifmo.ru/el_books/numerical_methods/lectures/glava3.html, 16.11.2017

3 Бронштейн И. Н., Семендяев К. А. Справочник по математике. - М.: Гос. изд-во техн.-теоретич. лит., 1956. – 608 с.

4 Димедович Б.П., Марон И. А., Шувалова Э.З. Численные методы анализа – М.: Наука, 1967. – 368 с.

5 Буч Г., Рамбо Дж. Джекобсон А. Язык UML. Руководство пользователя: Пер. с англ. - М.: ДМК, 2000. – 432 с.

6 Джим Арлоу, АйлаНейштадт. UML 2 и Унифицированный процесс. Практический объектно-ориентированный анализ и проектирование. – СПб.: Символ-плюс, 2007. – 624 с.

7 Марченко А. Л. C#. Введение в программирование.– М.: МГУ им. М.В. Ломоносова, 2005. – 317 с.

8 Троелсен Э.Язык программирования C#, и платформа .NET 4. 5-е издание. / Пер. с англ. - М.: ООО " И. Д. Вильямс", 2011. – 1392 с.

9 Печерский Александр Язык XML – практическое введение. – М.: Российское образование, 2002. – 312 с.