Дніпровський науковий ліцей інформаційних

технологій

Дніпровської міської ради

**Випускна робота**

**на тему:**

**Інтерполяція та апроксимація функцій**

**Виконавець:**

**ліцеїст 10А класу**

**Поєнко Михайло**

**Керівник роботи:**

**Боровик Л.І.**

**Дніпро**

**2021**

# Зміст

[Зміст 2](#_Toc89298940)

[Вступ 3](#_Toc89298941)

[Основна частина 4](#_Toc89298942)

[Теоретична частина 4](#_Toc89298943)

[Інтерполяція функцій 4](#_Toc89298944)

[Лінійна інтерполяція 5](#_Toc89298945)

[Квадратична інтерполяція 5](#_Toc89298946)

[Многочлен Лагранжа 6](#_Toc89298947)

[Многочлен Ньютона 6](#_Toc89298948)

[Апроксимація функцій 7](#_Toc89298949)

[Лінійна апроксимація 8](#_Toc89298950)

[Квадратична апроксимація 8](#_Toc89298951)

[Криві Безьє 10](#_Toc89298952)

[Програмна частина 11](#_Toc89298953)

[Інтерфейс програми 11](#_Toc89298954)

[Побудова 11](#_Toc89298955)

[Теорія 13](#_Toc89298956)

[Допоміжні процедури 15](#_Toc89298957)

[Методи інтерполяції 20](#_Toc89298958)

[Лінійна 20](#_Toc89298959)

[Квадратична 22](#_Toc89298960)

[Многочлен Лагранжа 24](#_Toc89298961)

[Многочлен Ньютона 27](#_Toc89298962)

[Методи апроксимації 30](#_Toc89298963)

[Лінійна 30](#_Toc89298964)

[Квадратична 32](#_Toc89298965)

[Криві Безьє 33](#_Toc89298966)

[Висновок 38](#_Toc89298967)

[Список використаних джерел 40](#_Toc89298968)

# Вступ

В математиці дуже важливу роль мають методи чисельного наближення функцій. Часто потрібно встановити залежність зміни якоюсь величини, за зробленими дослідами і отриманими даними. Іноді постає й інша задача, полегшити цю залежність для більш швидких розрахунків.

Тему роботу можна вважати актуальною, тому, що використання чисельних методів наближення функцій може слугувати апаратом при вирішення більш складних задач чисельного аналізу: інтегрування та диференціювання функцій, задача оптимізації та багато іншого.

Криві Безьє – це один з найважливіших та найбільш практичних інструментів у комп’ютерній графіці, з їх допомогою можливо побудувати гладкі криві будь якої форми, ці криві використовуються майже в усіх комп’ютерних шрифтах. Від їх використання не відмовилися такі векторні редактори, як Photoshop, Illustrator та Inkscape.

**Метою роботи** було розроблення програми, яка б могла виконувати побудову графіків деяких функції за дискретним набором значень, які задаються користувачем. Також обов’язковим пунктом було знаходження рівнянь побудованих функцій. Задачею також було розглянути велику кількість методів інтерполяції та апроксимації функцій і демонструвати побудову та роботу з кривими Безьє, реалізуючи їх алгоритми у програмному середовищі WinForms Visual Studio.

# Основна частина

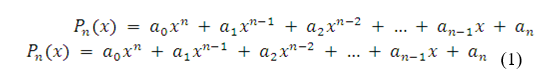
## Теоретична частина

### Інтерполяція функцій

**Інтерполяцією** функції вважаємо спосіб знаходження проміжних значень величини за наявним дискретним набором значень.

Изображение выглядит как текст

Автоматически созданное описаниеНехай на відрізку [a;b] задані значення функції у точках

Інтерполяція – це знаходження поліному, не вищого степеню за кількість обраних точок:

Який у точках приймає ті ж значення, що й вказана функція, тобто виконується рівність:



Інше кажучи, інтерполяція – це знаходження многочлену виду (1), який на відрізку [a;b] являв би собою приближення функції .

Многочлен (1) називаємо інтерполяційним многочленом, а точки

* вузлами інтерполяції.

У роботі було розглянуто 4 методи інтерполяції функцій:

* Лінійна
* Квадратична
* Многочлен Лагранжа
* Многочлен Ньютона

#### Лінійна інтерполяція

Найпростішим та найбільш популярним методом інтерполяції є **кусково-лінійна інтерполяція**. Вона полягає у тому, що через кожні дві точки (x1<x2<x3…) проведено прямі, тобто складається поліном першого ступеня, задачею було вирішення n-1 систем з двох рівнянь ввиду, де 0≤ i < n-1

Тобто задача зводилася до пошуку невідомих коефіцієнтів k та b для кожних двох точок та побудови функції виду

#### Квадратична інтерполяція

Ще одним методом інтерполяції є **кусково-квадратична інтерполяція**, цей метод полягає у побудові параболи для кожних трьох вершин інтерполяції (x0<x1<x2…), коефіцієнти шукаємо вирішуючи систему з трьох рівнянь.

, де 1≤ i<n-1

Отримуємо:

Задача зводиться до побудови функції виду для кожних трьох точок.

#### Многочлен Лагранжа

При глобальній інтерполяції одним многочлен на усій області визначення використовується **многочлен Лагранжа**, цей многочлен проходячи через вузли інтерполяції має дуже маленьку похибку за малих значень n. Многочлен Лагранжа є многочленом ступеня n-1. Многочлен Лагранжа часто використовують, коли вузли є незмінні, бо з додавання нового вузла потрібно знову вираховувати увесь многочлен. Многочлен Лагранжа має запис:

#### Многочлен Ньютона

Інтерполяційний многочлен Лагранжа іноді буває незручним, через те, що при додаванні нового вузла інтерполяції, потрібно перераховувати увесь многочлен заново, тому існує інша форма запису – **многочлен Ньютона**. Він виглядає так:

### Апроксимація функцій

Дуже часто для практичних розрахунків потрібно визначити функцію, зі значеннями якої могли б з високою точністю збігатися інші отримувані значення. Така задача називається **апроксимацією** функції, інтерполяція – це окремий вид апроксимації, коли функція проходить чітко через наявні точки.

Близькою є також задача **апроксимації** складної функції однією з простіших, використання такою функції звичайно не дає точних результатів, але іноді виграш у швидкості обчислень переважає похибку у отриманих результатах.

Найчастіше апроксимація використовується у дослідженнях, для обробки отримуваних значень для передбачення значень величини, важливо розуміти, що точних значень таким чином ми не отримуємо, але у нас є проста для розрахунків та використання функція, яка вказує на основну тенденцію зміни чогось.

Вид апроксимації потрібно вибирати враховуючи саме поставлені задачі досліду. У роботі реалізовано два способи апроксимації:

* Лінійна апроксимація
* Квадратична апроксимація

#### Лінійна апроксимація

Залежність між деякими даними дуже зручно та доречно можна описати прямою, тобто лінійною залежність виду

Коефіцієнти k та b можна знайти за **методом найменших квадратів**:

Изображение выглядит как текст

Автоматически созданное описание

Метод найменших квадратів

Звідки отримуємо:

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описаниеИзображение выглядит как текст

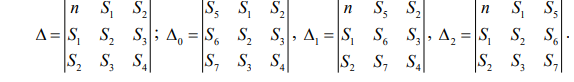
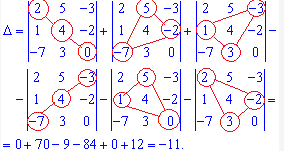
Автоматически созданное описаниеОтже коефіцієнти можемо обчислити за формулами

#### Квадратична апроксимація

Якщо лінійною функцією не вдається досить чітко відобразити набір даних, то використовують **квадратичну апроксимацію**, у такому випадку апроксимуюча функція представляє собою многочлен виду

Коефіцієнти ai цього многочлену вираховуються за МНК для многочлену 2 ст.

Изображение выглядит как часы

Автоматически созданное описаниеИзображение выглядит как текст, часы

Автоматически созданное описаниеИзображение выглядит как текст

Автоматически созданное описаниеИзображение выглядит как текст

Автоматически созданное описаниеИзображение выглядит как текст

Автоматически созданное описание

Ділення матриць робимо знайшовши й поділивши їх показники за поданою схемою заповнивши своїми значеннями

### Криві Безьє

Векторні зображення складаються з контурів. Контури складаються із сегментів, обмежених вузлами. З декількох таких сегментів можна скласти, практично, будь-яку фігуру. Для опису контурів у програмах векторної графіки застосовують розроблені французьким математиком П’єром Безьє параметричні поліноміальні криві. Відмітимо, що криві та поверхні Безьє були використані у шістдесятих роках компанією «Рено» для комп’ютерного проектування форми кузовів автомобілів. На сьогодні вони широко використовуються в комп’ютерній графіці, автоматизованих системах управління виробництвом тощо.

Крива Безьє являє собою поліноміальну, параметрично задану криву ввиду:

Де - базисні функції кривої Безьє, також відомі як поліноми Берштейна:

# Програмна частина

Під час розробки додатку використовувалась мова програмування С# та середовище розробки Visual Studio 2019.

Були додатково підключені простори імен:

* System.Drawing.Drawing2D
* System.IO

Використовувався клас Graphics та його методи DrawLine(), FillElipse(), DrawString() для малювання графіків та імен.

Використовувався клас Array та його методи Sort(), Resize() та Length для роботи з масивами точок.

Використовувався клас Math та багато його методів для підрахування значень функції.

Використовувалися класи Pen, Brush, SolidBrush та Color для надання кольору побудованим елементам.

Використовувався клас StreamReader для читання точок з текстового файлу.

Використовувався клас Font для створення шрифта.

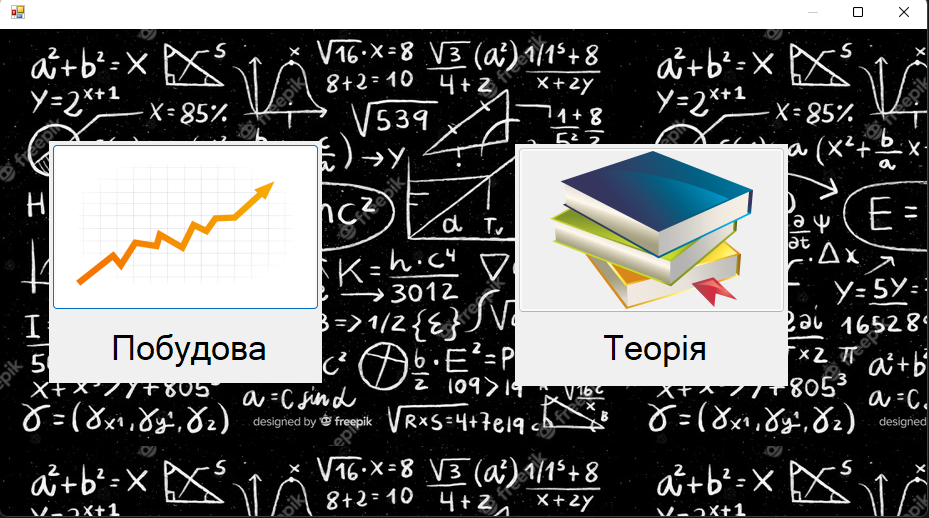
На форму додані такі елементи управління, як timer, для малювання кривих Безьє з анімацією. openFileDialog для читання вибраного файлу, colorDialog для вибору кольору, menuStrip для меню та toolTip для додання підказок.

На самій формі бачимо елементи управління button, label, dataGridView, panel, groupBox, textBox.

Увесь код поділен на #region.

### Інтерфейс програми

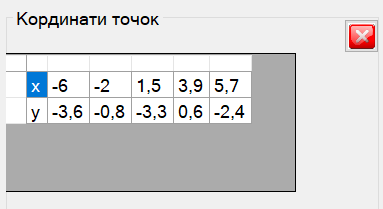
Під час виконання роботи було розглянуто 4 методи інтерполяції, 2 методи апроксимації та Криві Безьє.

**Інтерфейс** програми є дуже простим. Спершу користувач потрапляє на головну сторінку, де може вибрати один з двох розділів «Теорія» та «Побудова».

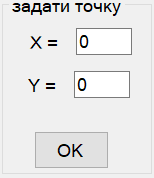
Головна сторінка

#### Побудова

На формі розташований pictureBox на якому відбуваються усі побудови. Зверху розташовано меню, за допомогою якого можна вибрати потрібний метод чисельного аналізу або вид відображення Кривої Безьє (елемент управління MenuStrip), за замовченням меню не є доступним одразу при запуску, на формі також розташовано ListBox, у який будемо виводити рівняння відповідних функцій. Біля PictureBox можемо побачити 4 кнопки, які відповідають за зсув системи в одну зі сторін, відповідно позначенню на кнопці, масштабування системи координат здійснюється на колесо миші. Також на формі розміщено 3 groupBox. За замовченням система будується при відкриті форми за розмірами від -10 до 10, але це можна змінити вибравши пункт меню Налаштування > Параметри системи координат і на панельці, що з’явилася вибрати бажані розміри.

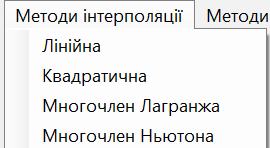
Другий groupBox є видимий за замовчуванням і являє собою елемент управління dataGridView, що заповнюється координатами створених точок.

groupBox2

Третій groupBox за замовченням є видимий і дозволяє користувачу вводити координати точок з клавіатури (іноді це може допомогти при роботі з дискретними даними).

groupBox3

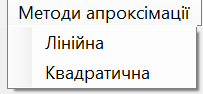
Розгляньмо меню програми, що складається з чотирьох основних пунктів:

1. *Методи інтерполяції*

За натисканням на даний пункт можна вибрати один із чотирьох доступних методів інтерполювання.

1. *Методи апроксімації*

ToolStrip1

За натисканням на даний пункт меню можна вибрати один із двох доступних методів апроксімації.

1. *Криві Безьє*

ToolStrip2

Изображение выглядит как текст

Автоматически созданное описаниеЗа натисканням за даний пункт меню можна вибрати одним з двох видів представлення кривої Безьє

ToolStrip3

1. *Налаштування*

Даний пункт дозволяє вибрати одну з чотирьох функцій:

1. *Параметри системи координат* – робить видимим groupBox1, для налаштувань системи координат
2. Вімк/вимкн вибір кольору – дозволяє вибрати один з двох режимів вибору кольорів:

* Вімк. – вибір кольору кожного графіка у діалозі
* Вимк – випадковий вибір кольору

1. *Изображение выглядит как текст

   Автоматически созданное описание*Підгрузити точки – дає можливість отримати координати точок з текстового файлу, може бути корисним для роботи з великої кількість даних

ToolStrip4



Panel 1

Представлена панелька відображає положення курсору на системі координат

Дана кнопка видаляє усі точки й графіки на системі

Видалення

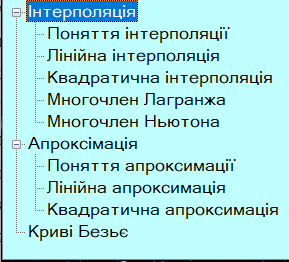
#### Теорія

Дана кнопка повертає вас до головного меню

Изображение выглядит как текст

Автоматически созданное описаниеДана форма дозволяє вам переглянути математичну частину та теорію, що була використана у роботі, не виходячи з проекту.

Форма складається лише з двох елементів:

1.treeView – на якому можна вибрати потрібний розділ роботи, для відображення теорії

2.WebBrawser – який відображає теорію за обраною темою, теорія подана лаконічно з додаванням потрібних формул

Изображение выглядит как текст

Автоматически созданное описание

### Допоміжні процедури

Изображение выглядит как текст

Автоматически созданное описание1.Об’явимо глобально певні змінні.

Код 1

Масиви **xt** та **yt** будуть сортуватися, потрібно для методів наближення, а **xn** та **yn** не будуть

Изображение выглядит как текст

Автоматически созданное описание2. Пропишемо скрипт, що буде виконуватися при відкриті форми

Код 2

**sys\_set()** – будує систему координат за дефолтними значеннями

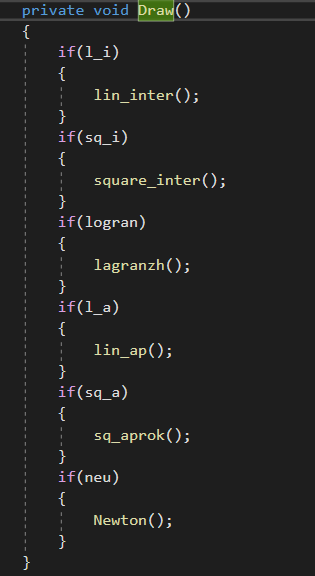
**Draw\_Points()** – будує вже додані точки (при минуломи сеансі на формі) (Код 3)

**Draw()** – будує вже побудовані графіки(при минуломи сеансі на формі) (Код 4)

Изображение выглядит как текст

Автоматически созданное описание

Код 3



Процедура викликає процедури побудови графіків використовуючи систему логічних змінних прописаних вище.

3. Пропишемо процедуру **AddPoint(x,y),** що могла б додавати до масивів координати заданих точок. (Код 5)

Изображение выглядит как текст

Автоматически созданное описаниеПроцедури **xtoi** та **ytoj** переводять коодинати точок із декартової системи у піксельну. Процедура **Sort()** сортує масиви **xt** та **yt** за зростанням, це потрібно для того, щоб забезпечити роботу деяких чисельних методів, що працюють лише з відсортованою інформацією.

Код 5

Код 4

1. Пропишемо процедуру, що могла б повертати систему у початковий стан. (Код 6)

Изображение выглядит как текст

Автоматически созданное описание

Код 6

1. Изображение выглядит как текст

   Автоматически созданное описаниеОпрацюємо подію натискання на pictureBox. (Код 7)

Код 7

1. Пропишемо процедуру **ColorPick()**, що повертає пезнель вибраного у діалоговому вікні кольору або випадковий колір, якщо подібне вибране в налаштуваннях. (Код 8)

Изображение выглядит как текст

Автоматически созданное описание

1. Додаємо заповнення таблиці за процедурою **FillTable()** (Код 9)

Изображение выглядит как текст

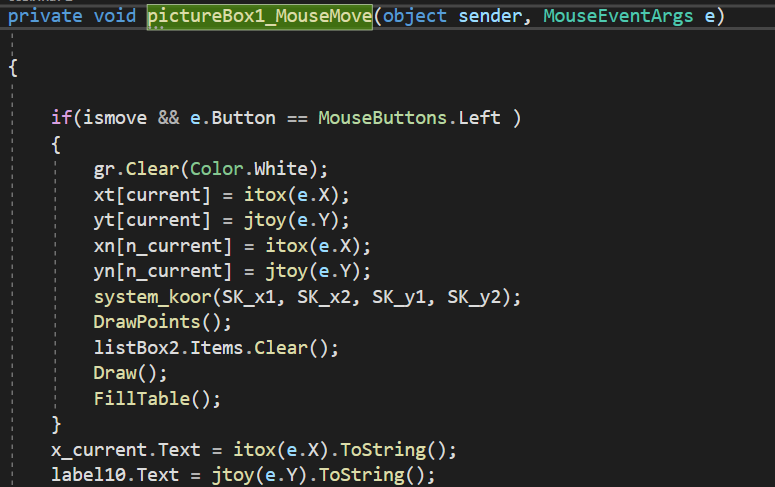
Автоматически созданное описание

Код 9

8. Додаємо можливість перетягування точок, опрацьовуючи подїї **MouseDown()** (Код 10) та **MouseMove()** (Код 11)

Изображение выглядит как текст, снимок экрана, экран

Автоматически созданное описание

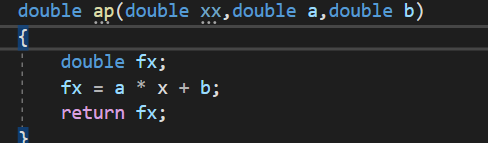


Код 10

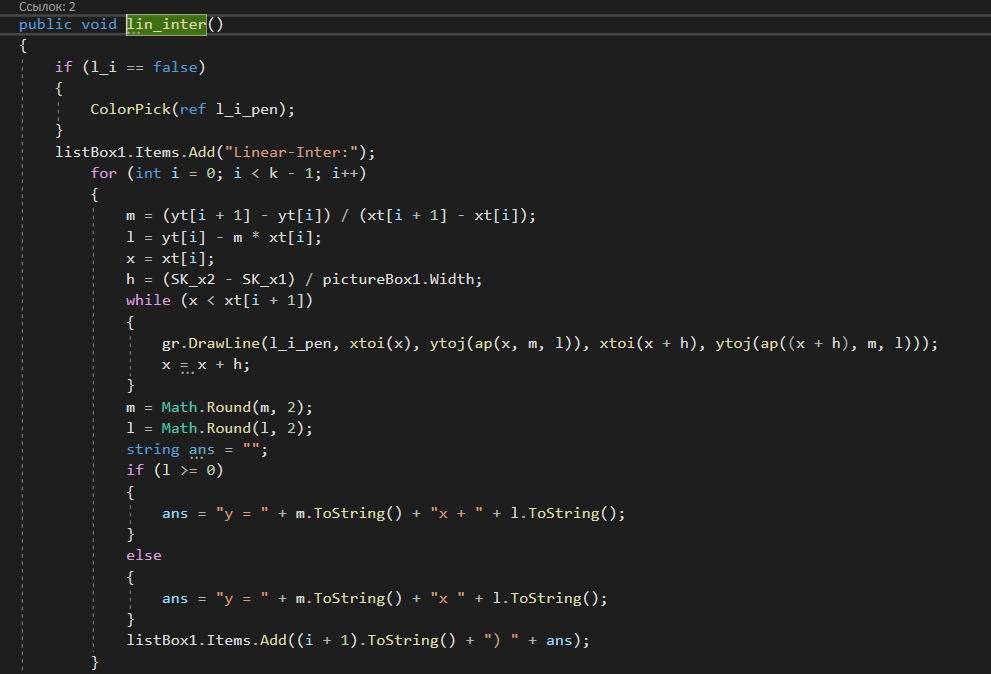
Код 11

### Методи інтерполяції

#### Лінійна



Знаходження f(x)



Знаходження коефіцієнтів, побудова функції та вивід рівняння прямої для кожних двох точок.

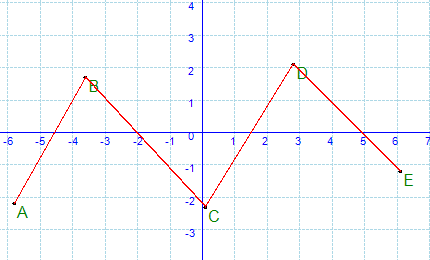
Розглянемо приклад побудови лінійної інтерполянти для випадкового набору точок:

1. Будуємо точки кліком на систему коорднат

2. Вибираємо у меню Методи інтерполяції – Лінійна інтерполяція

3. Вибираємо колір графіка у діалоговому вікні

4.Отримуємо:



Побудований графік вибраного кольору

Изображение выглядит как текст

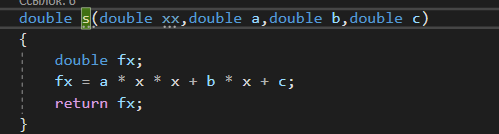
Автоматически созданное описаниеИзображение выглядит как стол

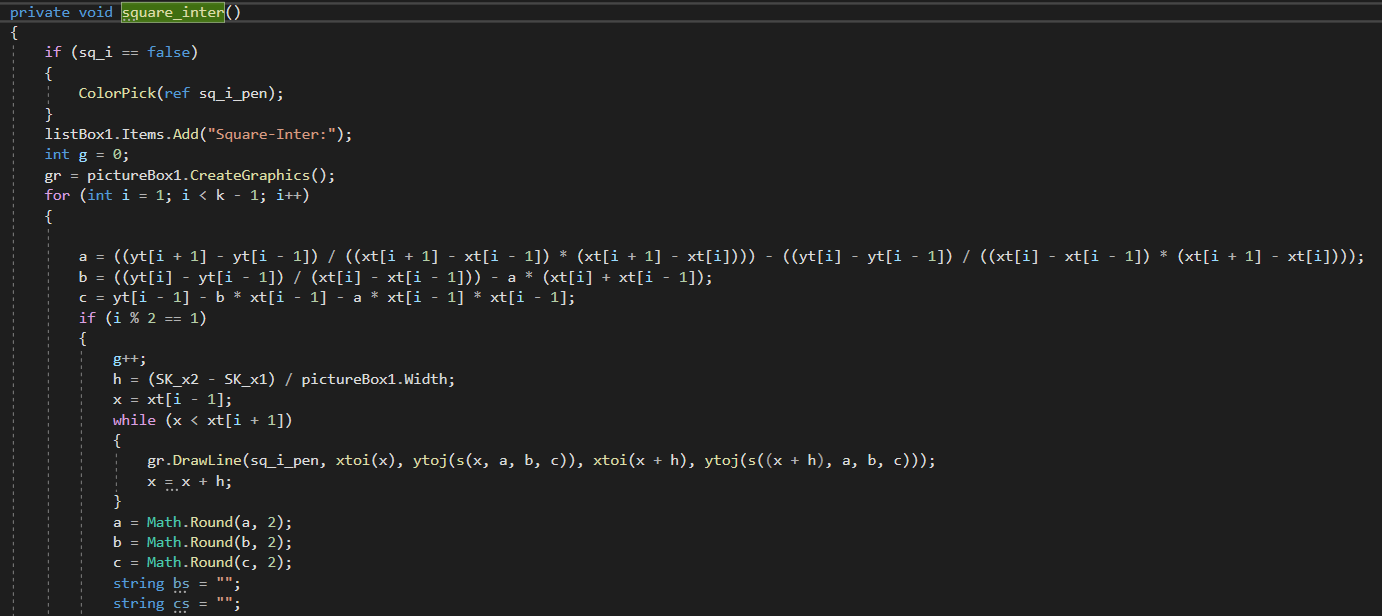
Автоматически созданное описание

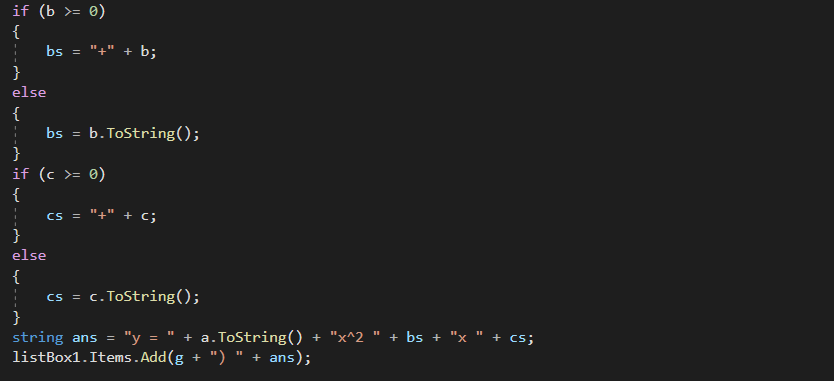
Рівняння прямих для кожних двох точок

Таблицю з координатами вибраних точок

#### Квадратична

**

Знаходження значення функції.



Знаходження коефіцієнтів, побудова функції та вивід рівняння параболи для кожних трьох точок.

Розглянемо приклад побудови квадратичної інтерполянти для випадкового набору точок:

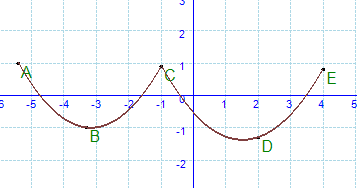
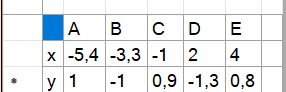
1. Будуємо точки кліком на систему координат

2. Вибираємо у меню Методи інтерполяції – Квадратична інтерполяція

3. Вибираємо колір графіка у діалоговому вікні

4. Маємо:

Изображение выглядит как текст

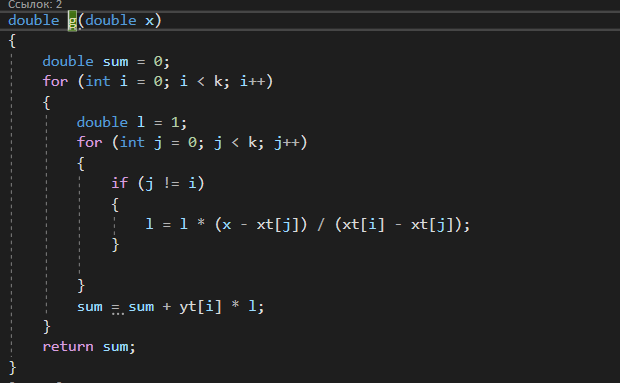
Автоматически созданное описание

Рівняння параболи для кожних трьох точок

Таблицю з координатами вибраних точок

Побудований графік вибраного кольору

#### Многочлен Лагранжа



Изображение выглядит как текст

Автоматически созданное описаниеФункція повертає значення поліному Лагранжу у певній точці.

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описаниеДані фрагменти коду шукають коефіцієнти многочлену Лагранжа та зберігає їх у масив.

Будування многочлену на системі та вибір кольору.

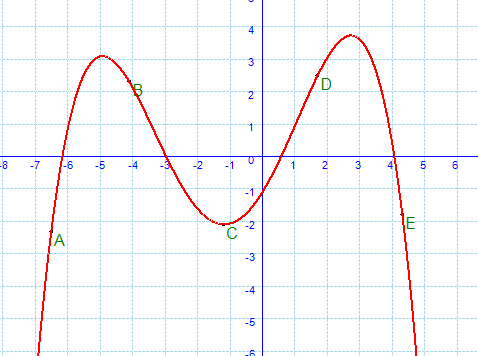
Розглянемо приклад побудови многочлена Лагранжа для випадкового набору точок:

1. Будуємо точки кліком на систему координат

2. Вибираємо у меню Методи інтерполяції – Многочлен Лагранжа

3. Вибираємо колір графіка у діалоговому вікні

4. Маємо:



Изображение выглядит как стол

Автоматически созданное описаниеИзображение выглядит как текст

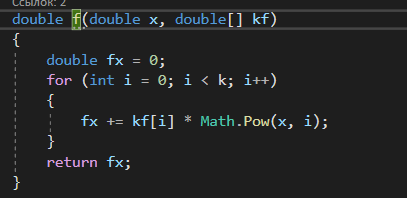
Автоматически созданное описание

Таблицю з координатами вибраних точок

Побудований графік вибраного кольору

Рівняння многочлену Лагранжа

#### Многочлен Ньютона



Функція для знаходження значення поліному Ньютона

Изображение выглядит как текст

Автоматически созданное описание

Дані фрагменти коду шукають коефіцієнти многочлену Ньютона та зберігають їх значення у двовимірний масив, також зберігається степінь аргумента при даному коефіцієнті

Изображение выглядит как текст

Автоматически созданное описаниеИзображение выглядит как текст

Автоматически созданное описание

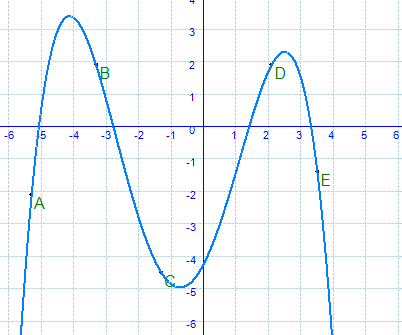
Фрагмент коду повертає строку, що відображає рівняння многочлену.

Розглянемо приклад побудови многочлена Ньютона для випадкового набору точок:

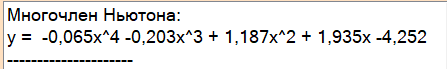
1. Будуємо точки кліком на систему координат

2. Вибираємо у меню Методи інтерполяції – Многочлен Ньютона

3. Вибираємо колір графіка у діалоговому вікні

4. Маємо:

Изображение выглядит как стол

Автоматически созданное описание

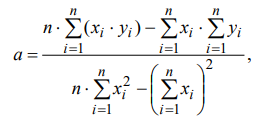
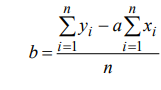
Таблицю з координатами вибраних точок

Рівняння Многочлену Ньютона

Побудований графік вибраного кольору

### Методи апроксимації

#### Лінійна

Коефіцієнти можемо обчислити за формулами

Изображение выглядит как текст

Автоматически созданное описание

Функція знаходить суми, необхідні для розрахунків.

Знаходження коефіцієнтів рівняння.

Изображение выглядит как текст

Автоматически созданное описаниеБудування прямої-апроксиманти та повернення її рівняння.

Розглянемо приклад лінійної апроксимації для випадкового набору точок:

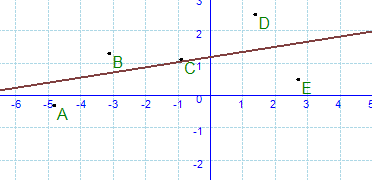
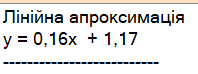
1. Будуємо точки кліком на систему координат

2. Вибираємо у меню Методи апроксимації – Лінійна

3. Вибираємо колір графіка у діалоговому вікні

4. Маємо:

Изображение выглядит как текст, легкий, закрыть

Автоматически созданное описание

Таблицю з координатами вибраних точок

Рівняння побудованої функції

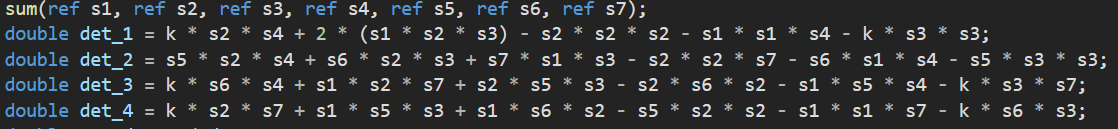
Побудований графік вибраного кольору

#### Квадратична



Изображение выглядит как текст, монитор, электроника, экран

Автоматически созданное описаниеПошук потрібних арифметичних сум.

Пошук показників вказаних матриць.

Изображение выглядит как текст

Автоматически созданное описаниеПошук коефіцієнтів параболи

Изображение выглядит как текст

Автоматически созданное описание

Виведення рівняння параболи.

Розгляньмо приклад побудови квадратичної апроксимуючої для випадкового набору точок:

1. Будуємо точки кліком на систему координат

2. Вибираємо у меню Методи апроксимації – Квадратична

3. Вибираємо колір графіка у діалоговому вікні

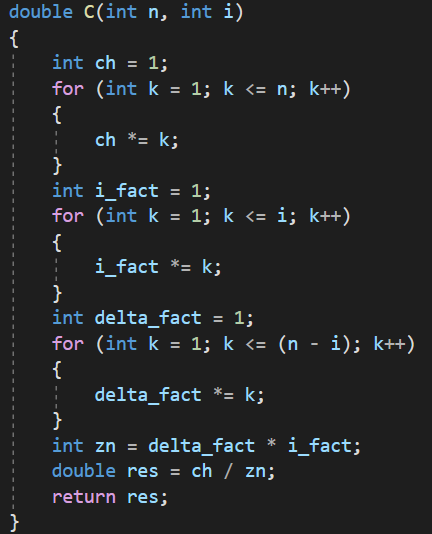
4. Маємо:

### Изображение выглядит как стол Автоматически созданное описаниеКриві Безьє

Таблицю з координатами вибраних точок

Рівняння апроксимуючої

Побудований графік вибраного кольору

Реалізація комбінаторної вибірки.

Изображение выглядит как текст

Автоматически созданное описаниеИзображение выглядит как текст

Автоматически созданное описаниеИзображение выглядит как текст

Автоматически созданное описаниеЗнаходження значень Кривої для усіх значень параметру t ( з кроком 0.02) та побудова сплайну.

Знаходження значення кривої Безьє у точці за параметром t

Реалізація поліному Берштейна

Изображение выглядит как текст

Автоматически созданное описаниеРеалізація анімації, оновленням зображення на кожний тік таймера.

Изображение выглядит как текст

Автоматически созданное описаниеprogressBar1 стає видимим лише при виборі відповідного пункту меню, також значення параметру виводимо на форму.

**bez\_limit() –** будує фрагмент кривої у відповідний момент часу.

**Изображение выглядит как текст

Автоматически созданное описаниеИзображение выглядит как текст

Автоматически созданное описаниеbz\_anim()** – будує точки, які утворюють криву у відповідний момент часу

Важливо зазначити, що масиви xn та yn не сортуються, бо за умовою вузли Кривої Безьє можуть бути невідсортовані, це сприяє утворенню петель, які використовуються наприклад у шрифтах.

Реалізація двох видів роботи обумовлено тим, що анімація може показати сутність кривої Безьє і легкість її побудови, а режим перетягування точок допомагає користувачу малювати контури.

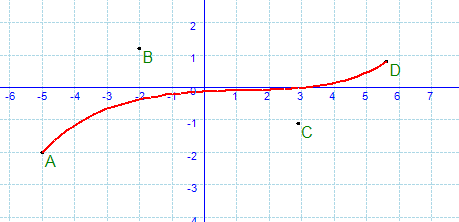
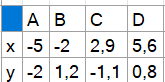
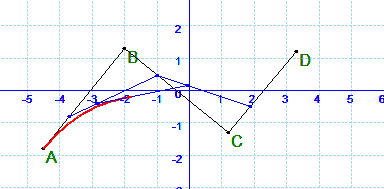
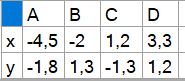
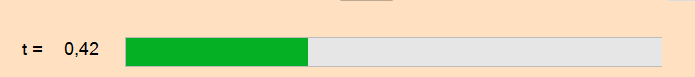
Розгляньмо приклад побудови квадратичної апроксимуючої для випадкового набору точок:

1. Будуємо точки кліком на систему координат

2. Вибираємо у меню Крива Безьє

3. Вибираємо один з двох режимів представлення

4. Маємо:

****

Значення параметру t

Вузли

Вузли

Крива Безьє у режимі анімації

Крива Безьє у режимі перетягування

# Висновок

Під час виконання курсової роботи було розроблено додаток у програмному середовищі С# Visual Studio, що допомагає користувачу задати точки на системі координат та побудувати інтерполянту або апроксимуючу функцію до певного набору точок. Також було розглянуто криві Безьє та реалізація їх побудови для необмеженої кількості точок з анімацією.

**Основною задачею роботи** була вдала реалізація основних методів чисельного наближення функцій та побудова кривої Безьє у коді програми та створення зрозумілого та зручного для користування інтерфейсу програми.

**Під час роботи було вивчено поняття** інтерполяції та апроксимації функцій за допомогою многочленів. Розглянуто 4 методи інтерполяції, серед яких два представляли собою графіки систем рівнянь і два формування многочлену, задля інтерполяції на усій області визначення функції. Також було засвоєно алгоритм побудови Кривих Безьє та розрахування значення параметричного поліному у певній точці.

**Найважчим кроком у реалізації інтерполяції** можна назвати визначення коефіцієнтів многочлену Лагранжа, бо знайдені початкові коефіцієнти потрібно змінити на фінальні, розкривши дужки у многочлені, ця задача була вирішена за допомогою використання бінарного коду для позначення доданків у дужках та лічильника, що позначає ступень доданку.

Насамперед було розроблено процедуру, яка додає координати певної точки до окремих масивів, сортує ці масиви та виводить координати у сітку. Також було розроблено процедуру, що дає можливість користувачу обирати колір побудованого графіку. Важкою задачею було забезпечення можливості зміни масштабу координатної сітки, результату було досягнуто методом використання процедуру побудови точок та системи логічних змінних, які запам’ятовують вибрані функції.

**Під час реалізації апроксимації функції я вивчив** метод найменших квадратів та основні його формули для двочленів першого та другого ступеня. Під час виконання апроксимації я навчився вирішувати системи рівнянь за методом Крамера та вдало реалізував це у системному коді програми.

Також було розглянуто поняття визначення показника матриці 2 на 2 та 3 на 3 та поняття ділення матриць.

**Реалізація побудови кривих Безьє** була досить складної, спершу я будував криву лише для 3 або 4 точок, варто зазначити, що саме такі криві використовуються у більшості випадків, але для повного засвоєння теми варто було створити можливість не обмежуватися певною кількістю точок. Найважчим кроком можна назвати створення наглядної анімації, яка б була приємною для ока звичайного користувача й при цьому відображала б побудову кривих.

**Було також оброблено деякі виключення**, серед яких спроба виконання одного з методів чисельного наближення функцій для однієї точки, що суперечить визначенню та невірний формат введених даних.

***У цілому код програми займає близько 15000 строк та сповна справляється з поставленою задачею, увесь код програми записаний у зрозумілих за призначенням процедурах. Також увесь код поділено на регіони для можливості комфортного пошуку та подальшого редагування. Усі функції наданні у проекті працюють без помилок.***

# Список використаних джерел

1. <https://portal.tpu.ru/SHARED/m/MBB/uchebnaya_rabota/Model/Tab/Interp_app.pdf>
2. <http://www.polybook.ru/comma/1.3.pdf>
3. <http://aco.ifmo.ru/el_books/numerical_methods/lectures/glava3.html>
4. <http://www.machinelearning.ru/wiki/index.php?title=%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BF%D0%BE%D0%BB%D1%8F%D1%86%D0%B8%D1%8F_%D0%BF%D0%BE%D0%BB%D0%B8%D0%BD%D0%BE%D0%BC%D0%B0%D0%BC%D0%B8_%D0%9B%D0%B0%D0%B3%D1%80%D0%B0%D0%BD%D0%B6%D0%B0_%D0%B8_%D0%9D%D1%8C%D1%8E%D1%82%D0%BE%D0%BD%D0%B0>
5. <https://studme.org/199289/informatika/lineynaya_approksimatsiya>
6. <https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BF%D0%BE%D0%BB%D1%8F%D1%86%D1%96%D1%8F>
7. <https://dlitdp.sharepoint.com/:b:/r/sites/-informatic-__2021/DocLib/%D0%A2%D1%80%D0%BE%D0%B5%D0%BB%D1%81%D0%B5%D0%BD%20%D0%AF%D0%B7%D1%8B%D0%BA%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_C%23_7_i_platformay_NET_i_NET_Core_(2018).pdf?csf=1&web=1&e=JDcYR9>