Черкаський національний університет імені Богдана Хмельницького

Кафедра програмного забезпечення автоматизованих систем

Курсова робота

з дисципліни «Програмування та алгоритмічні мови»

НА ТЕМУ «Візуалізація роботи алгоритму пошуку мінімуму функції однієї змінної методом парабол»

Студента 1 курсу, групи КС-19

напряму підготовки «Програмна інженерія»

спеціальності «Програмне забезпечення систем»

Слинько А.А

Керівник завідувач кафедри, доцент, кандидат фізико-математичних наук Онищенко Б. О.

Національна шкала: _____

Кількість балів: _____ Оцінка: ECTS _____

Члени комісії _____ Онищенко Б.О.

_____ Гребенович Ю.Є.

_____ Порубльов І.М.

м. Черкаси – 2020 рік

Зміст курсової роботи

Вступ

Актуальність. З розвитком сучасних технологій та збільшення обсягу поставленних задач людина шукає всі можливі способи полегшення виконання завдань. На допомогу з'являються інформаційні технології зі своїми безкрайніми можливостями. З'являються нові види програм для полегшення обчислювання різних задач. Зараз, такі програми, затребувані більше в фізико-математичних та технічних галузях. Наприклад: Математика, в якій існує дуже велика кількість формул і для розв'язування однієї важкої задачі людина витрачає досить багато часу. Фізика, в якій також досить багато формул для обчислення всих фізичних явищ. Металургія, в якій з приходом автоматизованої системи управління технологічним процесом, так званим АСУТП (рис. 0.1), збільшилася ефективність виробництва металопродукції.

Рис. 0.1. Панель управління АСУТП

На даний момент часу ми маємо дуже розвинуту сферу – інформаційні технології, за допомогою якої науковці можуть досить легко розв'язувати

поставленні перед ними задачі. Наразі дуже багато практичних задач зводиться до пошуку мінімума функції і для полегшення розв’язку таких задач за допомогою інформаційних технологій ми маємо готовий алгоритм.

Мета роботи. Продемонструвати користувачеві роботу алгоритму пошуку мінімуму заданої ним функції однієї змінної використовуючи метод парабол.

Завдання роботи:

1. Аналізувати алгоритми для операцій над функціями.
2. Побудувати блок-схеми алгоритмів програми.
3. Реалізувати програму, яка буде обраховувати функції та демонструвати її графік.
4. Зробити висновки щодо можливостей програмного продукту та аналізувати її застосування в людстві.

Базові поняття:

Алгоритм пошуку мінімуму – алгоритм для знаходження мінімальної точки в масиві.

Метод парабол – графік многочлена другого степеня.

Розділ 1. Огляд алгоритмів та методів

Розглянемо математичну галузь науки. На даний час є безліч програм, які можуть допомогти у вирішенню задач для математичних науковців. Однією з багатьох видів програм є вид програми, який допоможе візуалізувати роботу алгоритму пошуку мінімуму функції однієї змінної методом парабол. Цей алгоритм є найпростішим з усіх алгоритмів через те, що він не обмежує функцію і простий в реалізації. Один з багатьох класів функції, які легко апроксимуються, є многочлен другого порядку. Через те, що графік многочлена другого степеня виглядає так, як графік функції $\boldsymbol{x^2}$, а саме парабола, то такий метод зазвичай називають методом парабол. Такий метод є ефективним через те, що його результати при мінімізації гладких унімодальних функцій кращі за інші методи. Якщо є єдина точка $\boldsymbol{x^*}$, в якій $\boldsymbol{f(x)}$ набуває екстремального значення, то унімодальною функцією називається функція $\boldsymbol{f(x)}$, яка визначена на відрізку $[\boldsymbol{a, b}]$. Приклади унімодальних функцій наведені нижче (див. рис. 1.2).

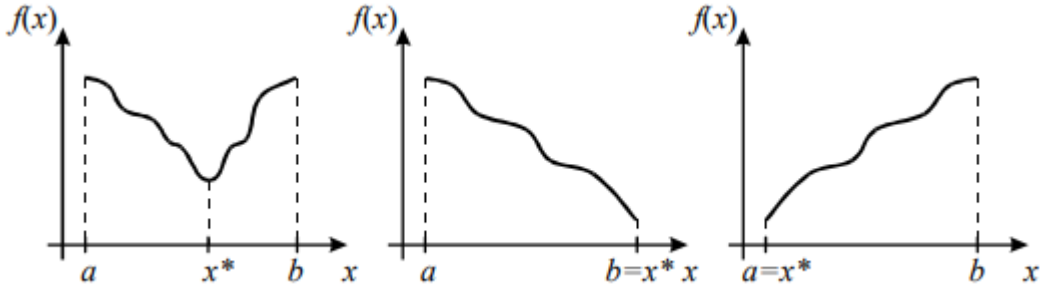


Рис. 1.2. Приклади унімодальних функцій

В основі багатьох методів закладено виключення з інтервалу $[\boldsymbol{a, b}]$ якої-небудь його не мінімальної частки. Мінімізація строго унімодальних функцій однієї змінної не потребує знаходження похідної. Для обрахунків задач існують наступні методи: метод дихотомії; метод золотого перетину; метод Фібоначчі та рівномірний пошук.

Метод парабол. В даній темі ж розглядається мінімізація гладких унімодальних функцій де найефективніше можна використати метод парабол. Метод апроксимації або ж так званий метод парабол відноситься до методів поліномальної апроксимації. Сама ідея цього методі базується на тому, що в деякій околиці мінімуму функції $\boldsymbol{f(x)}$ приймається точка мінімуму апроксимованого поліному. Через те, що апросимуюча функція є поліномом – мінімум легко знаходиться. Графічно цей метод виглядає так (див. рис. 1.7).

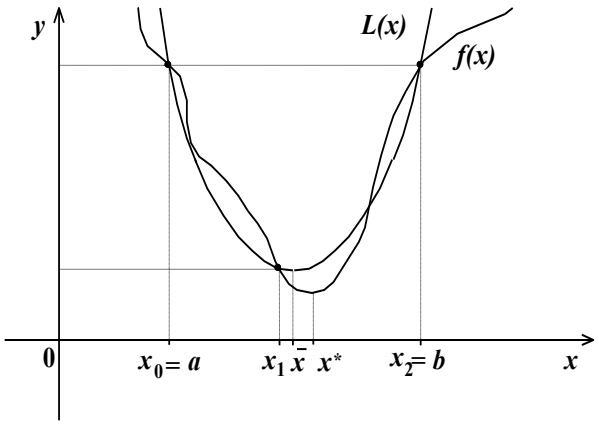


Рис. 1.7. Графічний вигляд методу парабол [3]

1.6 Висновок до першого розділу. В даному розділі розглядається актуальність теми в математичній галузі науки. Проводиться огляд алгоритму пошуку мінімуму функції та його переваги. Оглядається також метод парабол та його переваги.

Розділ 2. Опис роботи алгоритму та методу

2.1 Детальний опис методу парабол. В даній темі курсової роботи розглядається використовування методу парабол або ж метод апроксимації. Для того, щоб побудувати апроксимуючий многочлен другого степеня потрібно мати три точки $\boldsymbol{x_0, x_1, x_2}$, які задовільняють дану умову: $\boldsymbol{a = x_0 < x_1 < x_2 = b}$. Також потрібні функції $\boldsymbol{f(x)}$ в цих точках $\boldsymbol{f(x_0), f(x_1), f(x_2)}$. Використаємо інтерполяційний многочлен Лагранжа другого степеня для диференціованої форми який буде мати наступний вигляд (див. форм. 2.1):

$$\boldsymbol{L(x) = \left(\frac{f(x_2) - f(x_1)}{x_2 - x_1} + \frac{f(x_0) - f(x_1)}{x_1 - x_0}\right) * \frac{(x - x_1)(x - x_2)}{x_2 - x_0} + \frac{f(x_2) - f(x_1)}{x_2 - x_1}(x - x_1) + f(x_1)} \quad (2.1)$$

де $\boldsymbol{f(x_0), f(x_1), f(x_2)}$ *– це функції від заданих точок;*

$\boldsymbol{x_0, x_1, x_2}$ *– це задані точки.*

Існує означення яке говорить, що трійка чисел $\boldsymbol{x_0 < x_1 < x_2}$ називається вдалою, якщо вони задовільняють такі умови: $\boldsymbol{f(x_1) \leq \min\{f(x_0\}, f(x_2)\}}$ і $\boldsymbol{f(x_1) < \max\{f(x_0\}, f(x_2)\}}$. Остання умова означає, що точки $\boldsymbol{(x_0, f(x_0))}$, $\boldsymbol{(x_1, f(x_1))}, \boldsymbol{(x_2, f(x_2))}$ не паралельні осі абсцис (див. рис. 2.1). Із попереднього означення випливає, що точка $\boldsymbol{x^*}$ мінімуму функції $\boldsymbol{f(x)}$ міститься всередині наступного відрізку: $\boldsymbol{[x_0, x_2]}$.

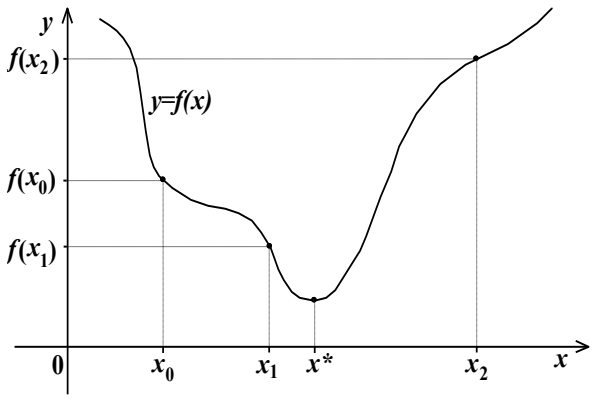


Рис. 2.1. Задані користувачем точки та функції від них не паралельні осі абсцис [3]

Якщо знайдена трійка яка задовілняє попередні умови, то випливає, що хоча б одна з наступних нерівностей: $\boldsymbol{f(x_1) \le f(x_0)}$ і $\boldsymbol{f(x_1) \le f(x_2)}$ вважається строгою і її коефіцієнт при старшому члені многочлена (див. форм. 2.1) додатній.

Визначивши похідну (див. форм. 2.2) многочлена якого ми знайшли раніше за допомогою формули інтерполяційного многочлена Лагранжа другого степеня для диференціованої форми (див. форм. 2.1) і прирівнявши її до нуля визначимо, що мінімум значення, якого ми знайшли раніше (див. форм. 2.1) досягається саме в наступній точці (див форм. 2.2).

$$\bar{x} = x_1 + \frac{1}{2}\frac{(x_2 - x_1)^2\big(f(x_0) - f(x_1)\big) - (x_1 - x_0)^2(f(x_2) - f(x_1))}{(x_2 - x_1)\big(f(x_0) - f(x_1)\big) + (x_1 - x_0)(f(x_2) - f(x_1)} \qquad (2.2)$$

де $\boldsymbol{f(x_0), f(x_1), f(x_2)}$ *– це функції від заданих точок;*

$\boldsymbol{x_0, x_1, x_2}$ *– це задані точки.*

при цьому $\frac{x_0+x_1}{2} \le \bar{x} \le \frac{x_1+x_2}{2}$.

Обирається точка для наступних обчислень значень функції $\boldsymbol{f(x)}$ (див. рис. 2.2), отримана з попередньої формули (див. форм. 2.2). Також, можливий випадок. коли $\overline{\boldsymbol{x}} = \boldsymbol{x_1}$ (див. рис. 2.3). Якщо ж таке сталось, то точка для наступних обчислення обирається з наступних формул: $\overline{\boldsymbol{x}} = \frac{\boldsymbol{x_0+x_1}}{\boldsymbol{2}}$ або $\overline{\boldsymbol{x}} = \frac{\boldsymbol{x_1+x_2}}{\boldsymbol{2}}$.

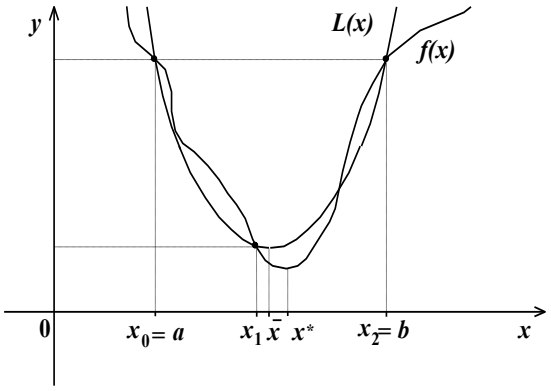


Рис. 2.2. Обирається точка для наступних обчислень значень функції $\boldsymbol{f(x)}$ [3]

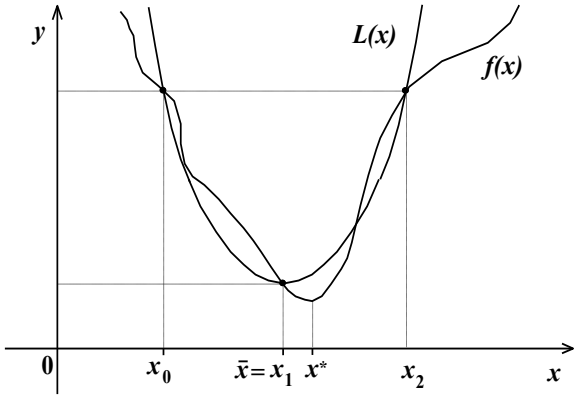


Рис. 2.3. Випадок $\overline{\boldsymbol{x}} = \boldsymbol{x_1}$ [3]

Після всих попередніх обчислень потрібно знову повторити обчислення за формулою (див. форм. 2.2), але уже з новою вдалою трійкою. Якщо наступна умова $|\boldsymbol{x_1} - \overline{\boldsymbol{x}}| < \boldsymbol{\varepsilon}$ виконується, то доцільно закінчити пошук наближеного значення точки мінімуму $\boldsymbol{x}^*$ і покласти при цьому $\boldsymbol{x}^* \approx \overline{\boldsymbol{x}}, \boldsymbol{f(x^*)} \approx \boldsymbol{f(\overline{x})}$. Коли

відомий відрізок локалізації мінімуму стає досить малої довжини, то правильним буде використанням методу парабол. Схожий цьому відрізку можемо отримати після $\boldsymbol{k}$ кроків методу дихотомії або методом золотого перерізу про які говорилось в попередньому розділі.

2.2 Блок схема до методу парабол. Блок-схема методу парабол (див. рис. 2.4) працює наступним чином. Запрошуються дані від користувача, а саме: функція, проміжок на осі X та точність мінімуму. Далі присвоюється значення функцій $\boldsymbol{f(x)}$ від точок $\boldsymbol{x_0, x_1, x_2}$. Після цього обчислюється значення точки $\boldsymbol{x^*}$ та функції від цієї точки $\boldsymbol{f(x^*)}$. Потім йде умова: поки різниця точок $\boldsymbol{x_1, x_2}$ та точки $\boldsymbol{x^*}$ більша або дорівнює точності мінімуму, то виконується наступна перевірка. Якщо точка $\boldsymbol{x^*}$ менша за точку $\boldsymbol{x_1}$, то виконується переприсвоєння точок $\boldsymbol{x_0, x_1, x_2, x_3}$ та точки $\boldsymbol{x^*}$ та їх функцій $\boldsymbol{f(x)}$. Якщо ж навпаки точка $\boldsymbol{x^*}$ більша за точку $\boldsymbol{x_1}$, то виконується переприсвоєння тільки точок $\boldsymbol{x_2, x_3}$ та точки $\boldsymbol{x^*}$ та їх функцій $\boldsymbol{f(x)}$. Все це зводиться в наступну перевірку. Якщо функція від точки $\boldsymbol{x_1}$ більша за функцію від точки $\boldsymbol{x_2}$, то виконується переприсвоєння точок $\boldsymbol{x_0, x_1, x_2, x_3}$ та їх функцій $\boldsymbol{f(x)}$. Якщо ж навпаки точка функція від точки $\boldsymbol{x_1}$ менша за функцію від точки $\boldsymbol{x_2}$, то виконується превірка на правильность і якщо все правильно, то програма виводить результат. Але якщо не правильно, то програма вертається до обчислення значення точки $\boldsymbol{x^*}$ та функції від цієї точки $\boldsymbol{f(x^*)}$ та повторюється все заново.

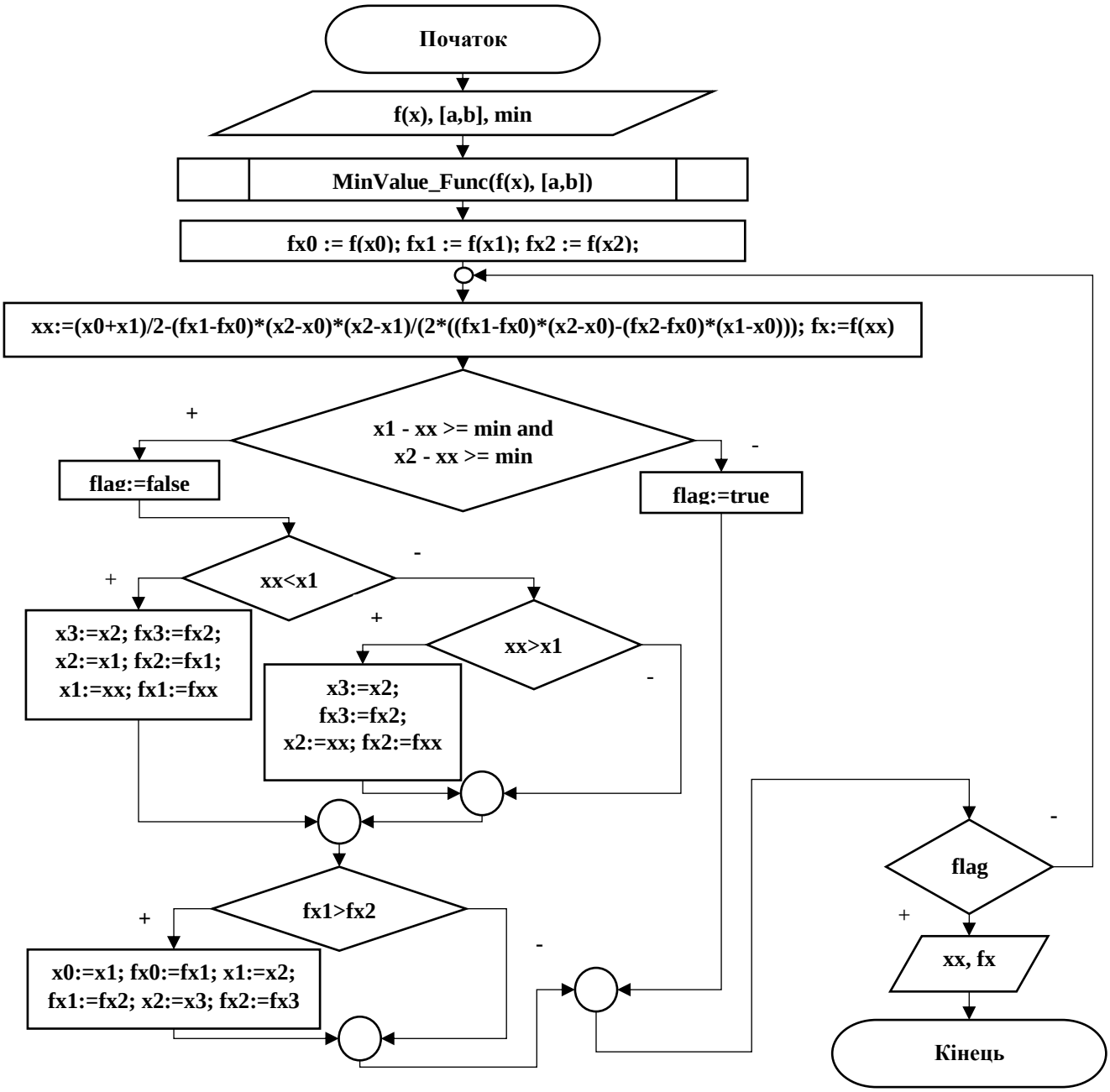


Рис. 2.4. Блок-схема методу парабол

Блок-схеми допоміжних алгоритмів зображені у додатках (див. додатки А-Г)

2.3 Висновок до другого розділу. У цьому розділі наведений детальний опис методу парабол. Детальний опис побудови апроксимуючого многочлена другого степеня. Опис блок-схеми методу парабол та сама блок-схема.

Розділ 3. Опис програми та її функціоналу

3.1 Короткий опис програми. Основа даної програми – це будувати графіки за заданої користувачем функції та відображати ці графіки на проміжку заданому користувачем. Також, невід'ємною частиною, є те, що користувачу потрібно ввести значення точності мінімуму для необхідних обчислень.

Програма має досить простий інтерфейс та легка у використанні. Перше, що бачить перед собою користувач це вікно (див. рис 3.1) в якому вже є необхідні інструкції, а саме: у TextBox під номером «1» потрібно ввести функцію за якою буде будуватись графік. У TextBox під номером «2» нехобхідно ввести проміжок на осі X на якому буде зображений цей графік. TextBox під номером «3» служить для того, щоб отримати значення точності мінімуму. Кнопка під номером «4» потрібна, щоб запустити процес будування графіку, функції та відповідних точок. Кнопка під номером «5» - стирає поле для малювання та видаляє дані в попередніх трьох TextBox.

Рис. 3.1. Початкове вікно програми

3.2 Особливості введення функції. Так, як програма не доскональна, то на даному етапі є деякі обмеженості. Для найпростіших функцій необхідне використування букв, а саме англійської букви «X», тут і постає проблема, як перетворити «X» в число. На допомогу приходить метод CultureInfo.CreateSpecificCulture(). В програмному коді він виглядає наступним чином (див. рис. 3.2), де equation.Text – це TextBox під номером «1» (див. рис. 3.1), в якому користувач вводить потрібну йому функцію, а .Replace() – це метод який вертає змінений рядок.

```
string str = equation.Text.Replace("x", x.ToString("F", CultureInfo.CreateSpecificCulture("en-US")));
```

Рис. 3.2. Вигляд методу CultureInfo.CreateSpecificCulture() у програмному коді

Для реалізації введеної користувачем інформації використовується метод DataTable().Compute для виконання операцій над заданою функцією. В програмному коді цей метод виглядає наступним чином (див. рис. 3.3). Даний метод використовується для виконання простих операцій над числами, а саме: додавання чисел (знак «+»), віднімання чисел (знак «-»), множення чисел (знак «*») та ділення чисел (знак «/»). Так, як даний метод не має знаку степеня (знак «^») його можна, і було б навіть логічним, замінити на множення (знак «*»).

```
double fx = Convert.ToDouble(new DataTable().Compute(str, ""));
```

Рис. 3.3. Вигляд методу DataTable().Compute у програмному коді

3.3 Особливості введення проміжку. Тут все доволі просто. Щоб правильно задати проміжок на вісь X потрібно ввести два натуральних числа

розділивши їх пробілом. Де перше число – це мінімальне значення проміжку на осі X, а друге число – максимальне значення проміжку на осі X. Вся ця операція супроводжується методом .Split(), який використовується для розкладання рядку з роздільниками на підрядки. Цей метод у програмному коді виглядає наступним чином (див. рис 3.4). Де num – це значення TextBox під номером «2» (див рис. 3.1), а саме проміжок який вводить користувач.

```
string[] str = num.Split(' ');
```

Рис. 3.4. Вигляд методу Split() у програмному коді

3.4 Особливості введення значення точності мінімуму. Значення точності мінімуму введене користувачем служить для того, щоб знайти мінімальні значення у функції та побудувати відповідні точки на графіку і осях. В TextBox під номером «3» (див. рис. 3.1) потрібно ввести одне не від'ємне число.

3.5 Призначення кнопок в програмі. Інтерфейс даної програми має лише дві кнопки. Кнопка під номером «4» (див. рис. 3.1) служить для того, щоб взяти необхідну для обчислень інформацію введену користувачем раніше. Обчислити межі графіка, а саме обмежити вісь абсцис до проміжку вказаним користувачем та обмежити вісь ординат до максимального і мінімального значення заданої функції. Обчислити функцію та побудувати її графік. Обрахувати мінімальні значення функції та побудувати відповідні точки на графіку. Кнопка ж під номером «5» (див. рис. 3.1) має менш обширний функціонал. Ця кнопка служить для того, щоб очистити поле для малювання графіку від самого графіку та функції намальованої попередньо за допомогою методу .Series[].Points.Clear(). Вигляд цього методу у програмному коді виглядає наступним чином (див. рис. 3.5). Де graph – це клас PictureBox. Клас

PictureBox – являє собою елемент управління вікном для відображення зображення.

```
graph.Series[0].Points.Clear();
```

Рис 3.5. Вигляд методу .Series[].Points.Clear() у програмному коді

Кнопка під номером «5» (див. рис. 3.1) також витирає данні введені користувачем в полях, де потрібно ввести функцію, проміжок і значення точності мінімуму. На цьому функціонал цієї кнопки закінчується.

3.6 Поле для відмальовки. В даній програмі використовується метод .CreateGraphics(), який допомагає створити свій графік з заданими величинами. Вигляд цього методу у програмному коді виглядає наступним чином (див. рис. 3.6). Де graph – це клас PictureBox.

```
Graphics g = graph.CreateGraphics();
```

Рис 3.6. Вигляд методу .CreateGraphics() у програмному коді

Далі, щоб побудувати осі, використовується метод .DrawLine, який служить для того, щоб провести лінію між двума точками з заданими координатами. Щоб провести лінію необхідно мати клас Pen, який задає колір, та ширину самої лінії, також необхідно мати структуру Point, яка в свою чергу створює точку з заданими координатами X та Y. Таких структур повинно бути дві, так як для проведення однієї лінії повинні дві точки. Приклад використання методу .DrawLine (див. рис. 3.7). Де g – це поле на якому малюється сам графік, for_axis – це клас Pen та new Point() – це дві точки з заданими координатами X та Y.

```
g.DrawLine(for_axis, new Point(0, num_s * (num_c / 2)), new Point(graph.Width, num_s * (num_c / 2)));
```

Рис. 3.7. Використання методу .DrawLine в програмному коді

3.7 Висновок до розділу. В даному розділі було детально розглянуто особливості інтерфейсу. Було проведено короткий інструктаж по використанню даної програми. Розглянуто, як саме працює програма та які функції вона виконує. Описані основні інструменти програми, які використовуються для вирішення поставленої задачі.

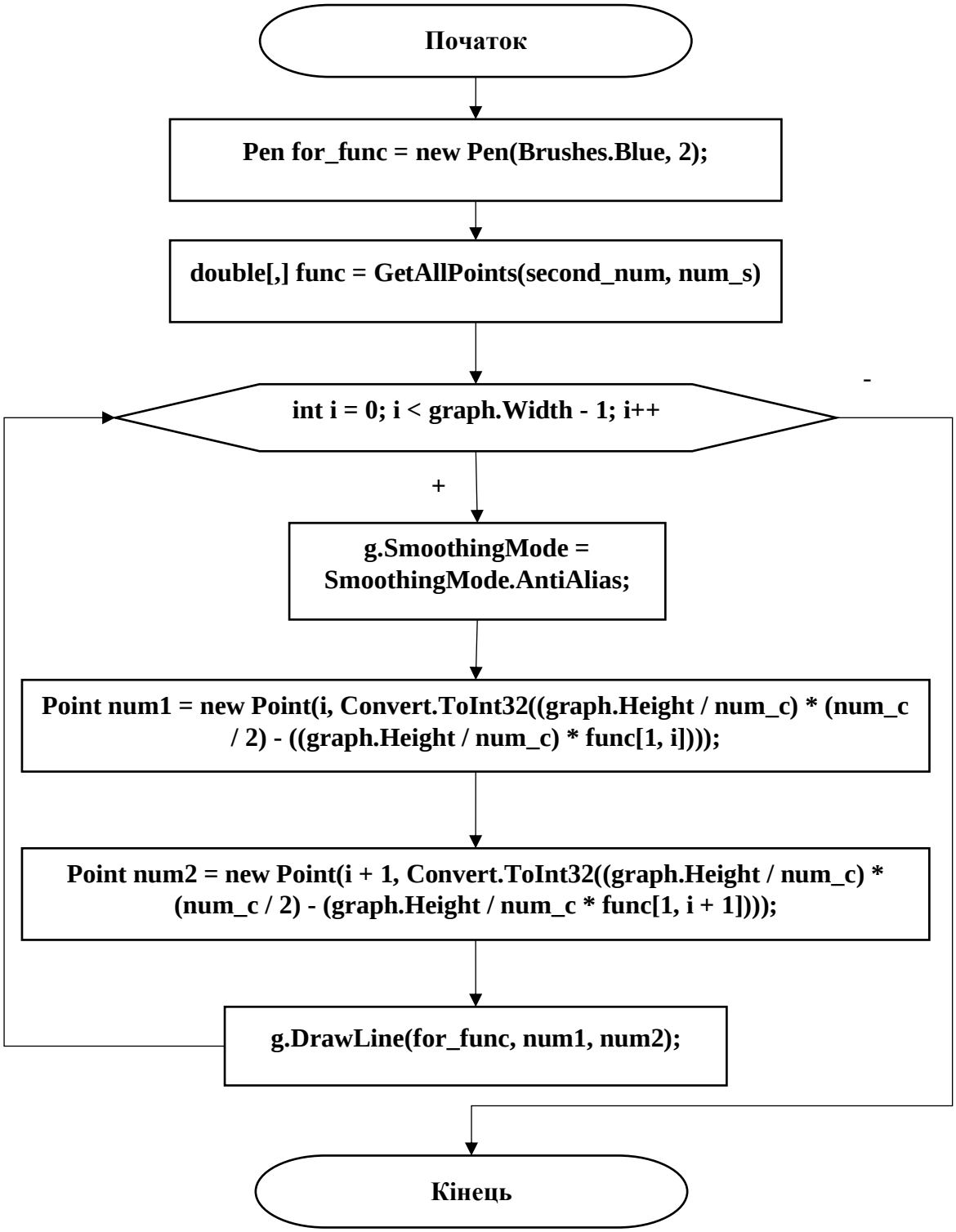
Висновки

Функціонал програми програми, яка була створена для розв'язувань певних математичних задач дуже простий і легкий у використанні. Даний програмний продукт може широко використовуватись в фізико-математичних галузях науки. Для полегшення розв'язувань різних задач тісно пов'язаних з графіками функцій. Маючи певні дані, а саме: функція $\boldsymbol{f(x)}$, проміжок та значення точності мінімуму, програма може легко розв’язати поставлену задачу.

Список використаних джерел

1. Microsoft Docs System.Windows.Forms Namespace [Електронний документ]. / Режим доступу: https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms?view=netcore-3.1 Перевірено: 21.05.2020
2. Mathros [Електронний документ]. / Режим доступу: http://www.mathros.net.ua/minimizacija-funkcii-odnijei-zminnoi-metodom-rivnomirnogo-poshuku.html Перевірено: 21.05.2020
3. StudFiles [Електронний документ]. / Режим доступу: https://studfile.net/preview/5457152/page:3/ Перевірено: 21.05.2020
4. Algolost [Електронний документ]. / Режим доступу: http://algolist.manual.ru/ Перевірено: 21.05.2020
5. Habr [Електронний документ]. / Режим доступу: https://habr.com/ru/search/ Перевірено: 21.05.2020
6. РЕФ.РФ [Електронний документ]. / Режим доступу: http://referatwork.ru/category/matematika/view/589032_metod_zolotogo_secheniya Перевірено: 21.05.2020
7. Bpascal [Електронний документ]. / Режим доступу: http://bpascal.ru/download/desc/319.php Перевірено: 21.05.2020

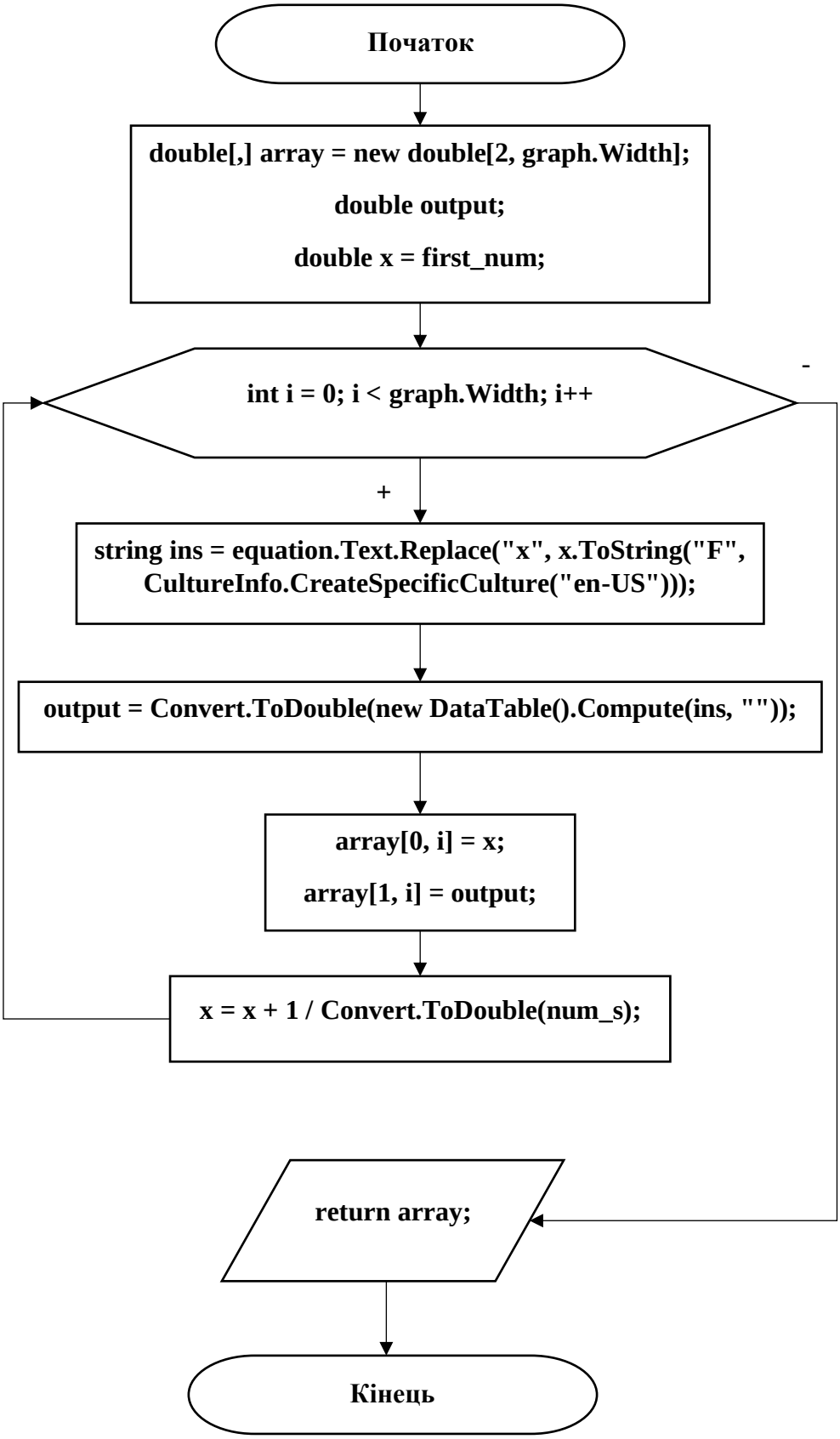
Додаток А. Блок-схема відмальовки функції



В даній блок-схемі розглядається метод відмальовки функції заданої користувачем. Спочатку створюється клас Pen для того, щоб задати колір функції. Потім беруться точки знайдені в наступному методі (див. Додаток Б.

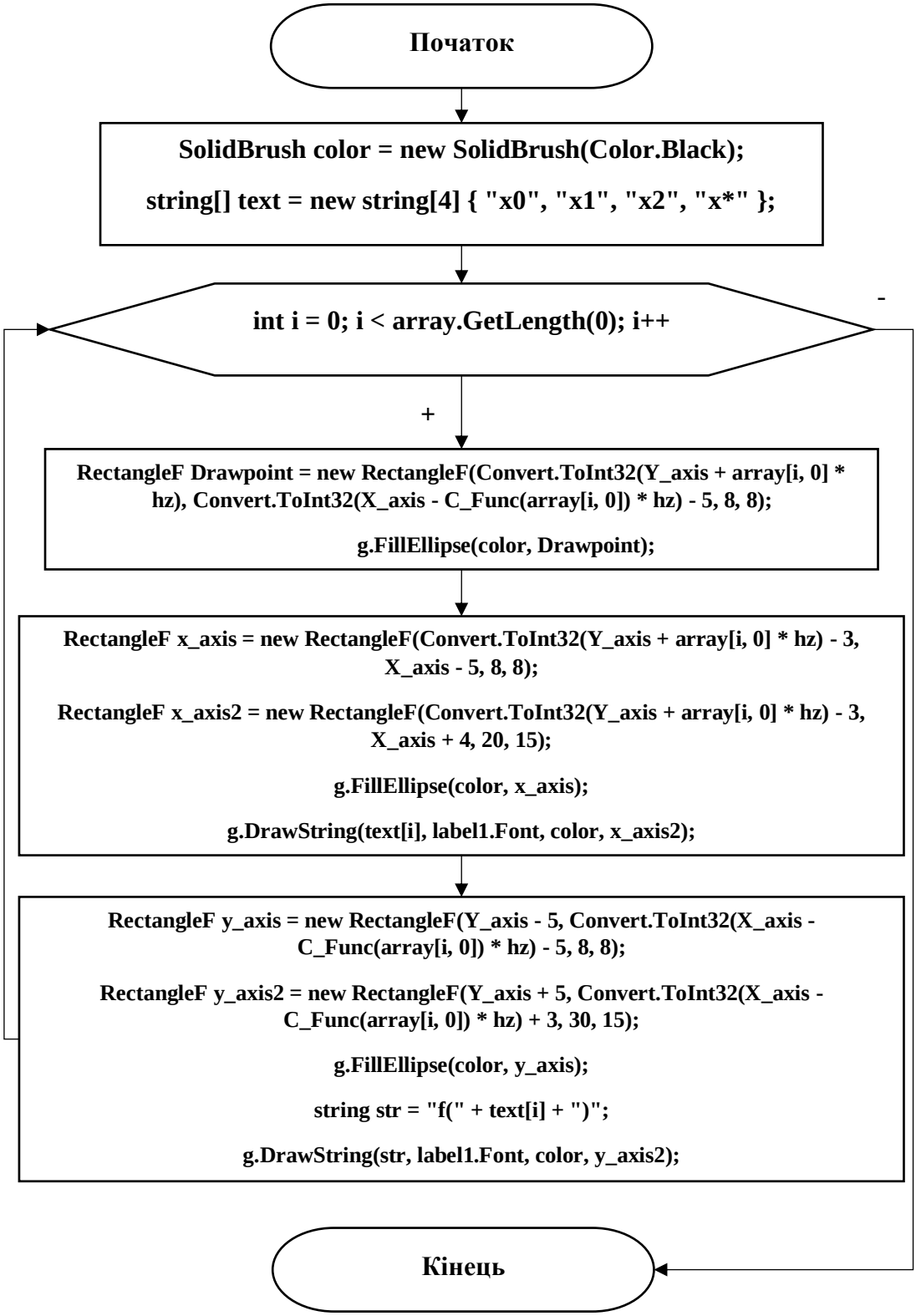
Метод знаходження того для функції) та створюється цикл довжиною від 0 до ширини самого поля для відмальовки графіку, але віднімаючи на один піксель щоб індекс не виходив за межі масиву. В самому ж циклі оптимізується якість відмальовки функції, створюється та обраховується перша та друга частина графіку функції $\boldsymbol{f(x)}$ та малюється на полі для відмальовки графіків.

Додаток Б. Знаходження точок для функції



В даній блок-схемі розглядається метод знаходження точок для функції. Спочатку створюється двумірний масив, після нього створюється змінна output для подальшого призначення результату, присвоюється значення першого числа проміжку змінній та створюється цикл довжиною від 0 до ширини самого поля для відмальовки графіку. В циклі перетворюється введене в функцію значення X на число та щоб виконувати математичні операції над цим значенням присвоюємо нове значення до нової змінної з методом DataTable().Compute, який саме і допоможе виконувати ці операції. Далі присвоюється два значення двумірного масиву, проводяться певні операції зі змінною якій присвоєно значення першого числа на проміжку та виходячи з циклу вертаємо значення масиву.

Додаток В. Відмальовка точок які були знайдені методом парабол



В даній блок-схемі представлений метод відмальовки точок, які були знайдені методом парабол. Спочатку створюється клас SolidBrush для того, щоб задати колір всим точкам та буквам, які відмальовуються в даному методі. Створюється масив з довжиною в 4 символи, ним ми створюємо букви для відображення на графіку. Далі цикл з розміром від 0 до розміру масиву який було створено раніше. В циклі ми обраховуємо координати і малюємо точки на осі абсцис, осі ординат та точки які належать функції. Також малюємо букви під точками, які створили тільки що.