Лабораторна робота №5

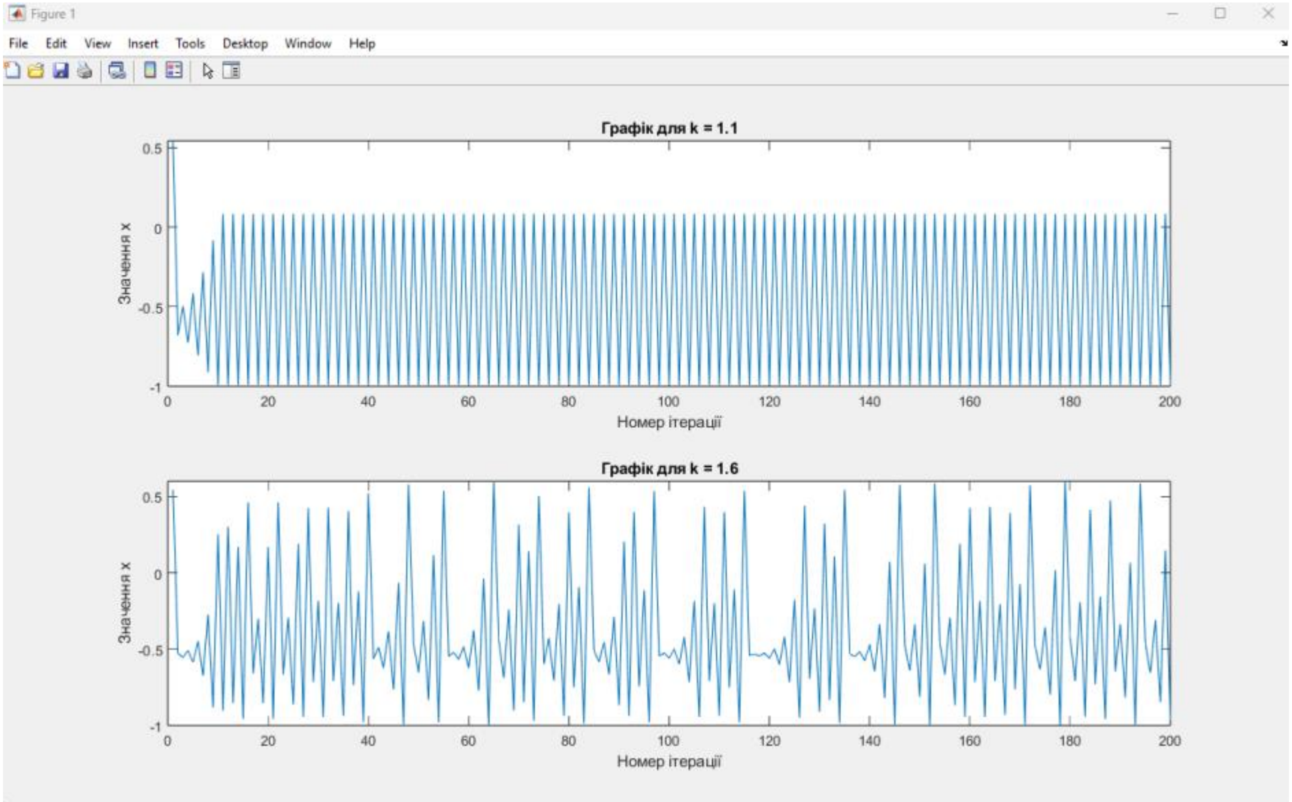
Студента групи КН-11

Сеня Тараса

З дисципліни Комп'ютерна графіка

Виконання

Приклад 5.1



Приклад 5.2

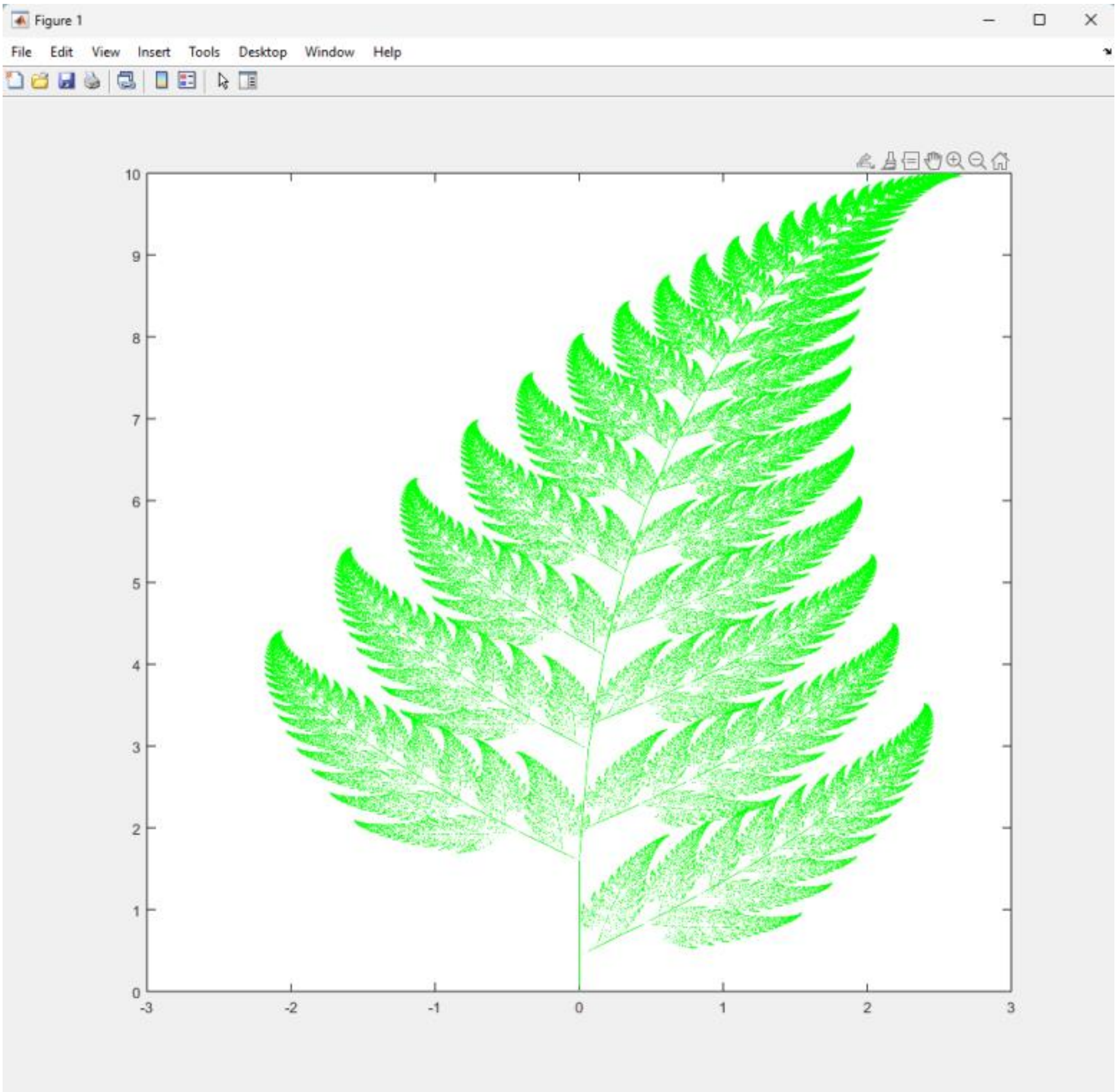


Figure 1
File Edit View Insert Tools Desktop Window Help
10
9
8
7
6
5
4
3
2
1
0
-3
-2
-1
0
1
2
3

Приклад 5.3

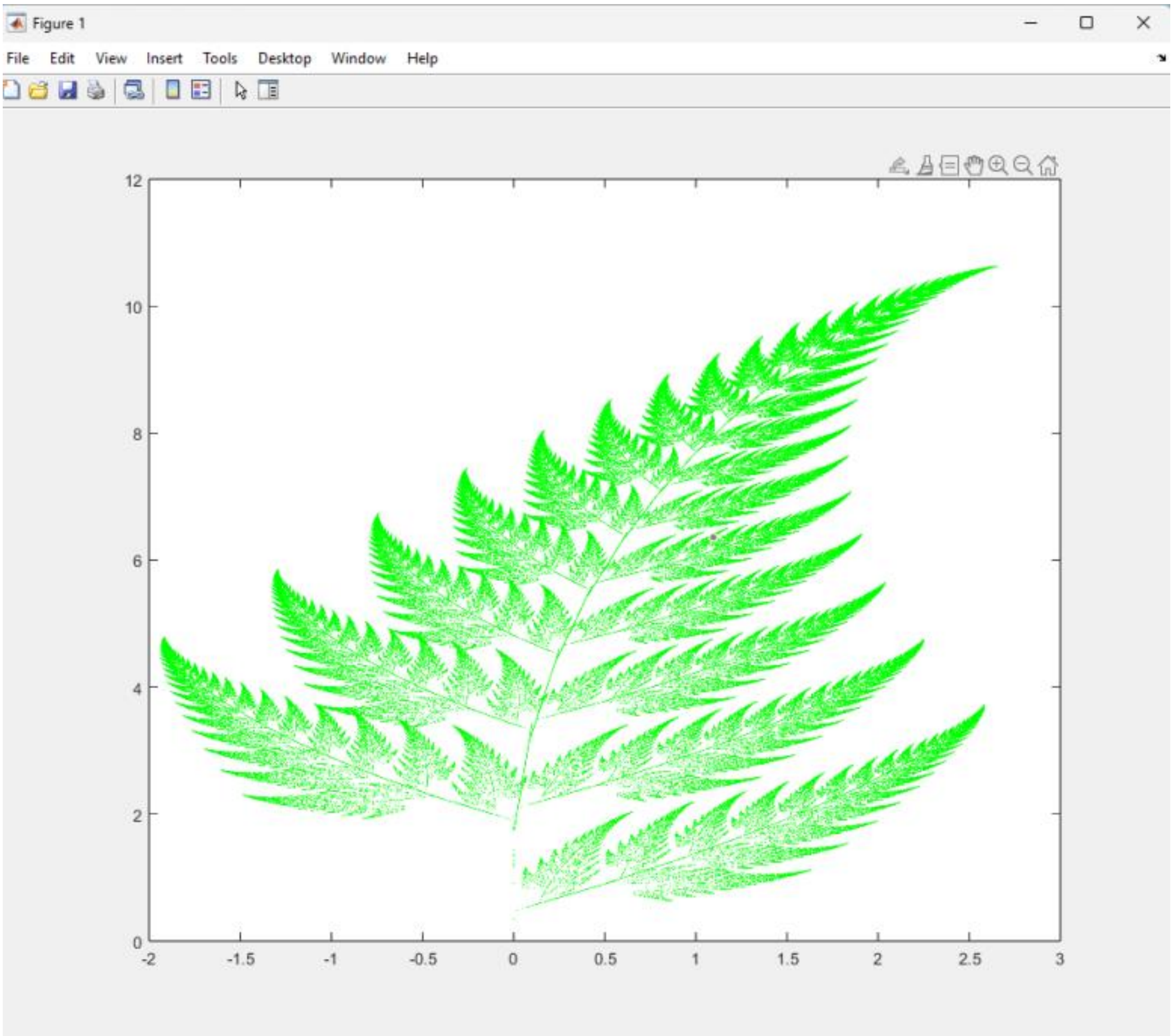


Figure 1
File Edit View Insert Tools Desktop Window Help
12
10
8
6
4
2
0
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
3

Приклад 5.4

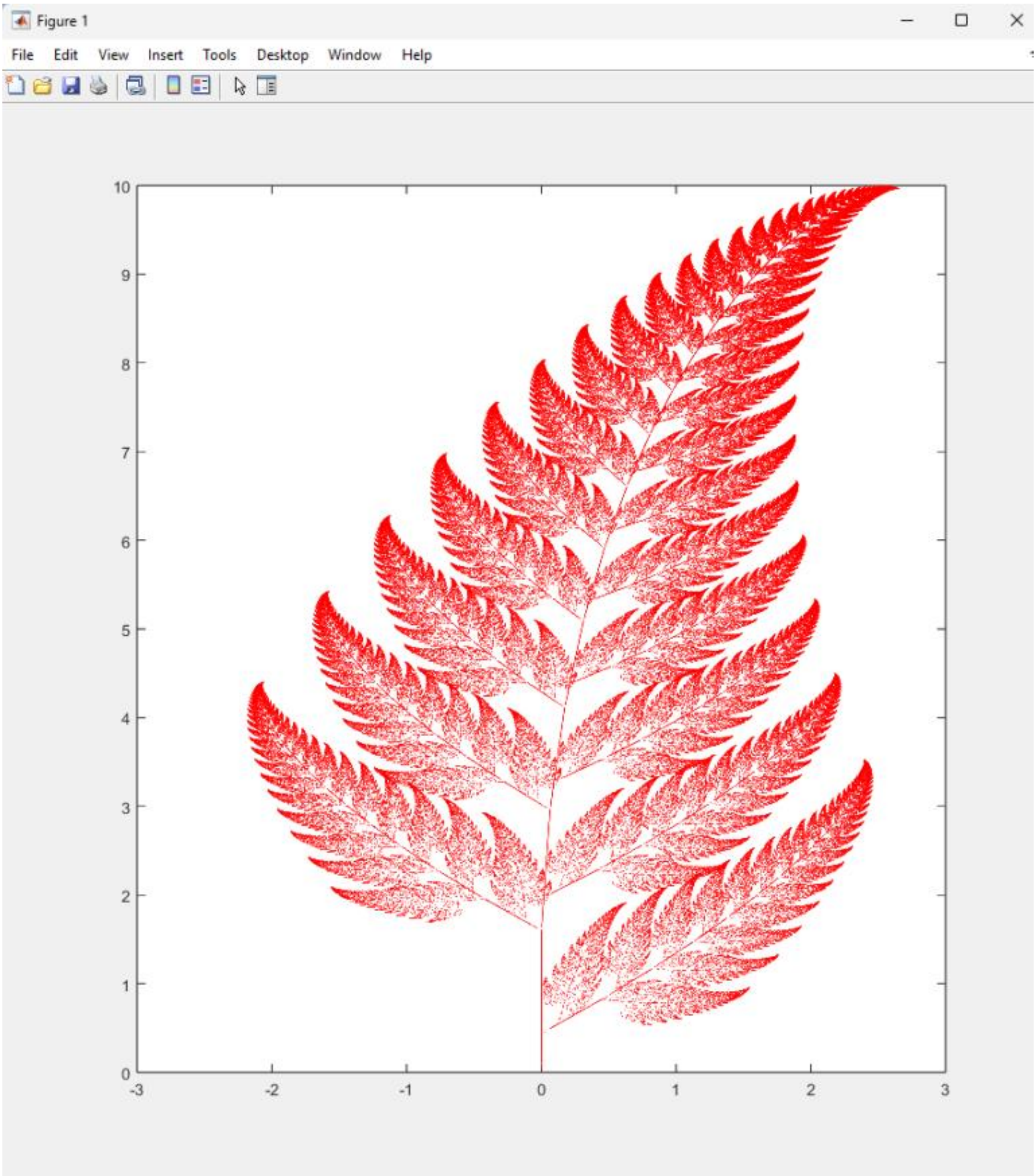


Figure 1
File Edit View Insert Tools Desktop Window Help
10
9
8
7
6
5
4
3
2
1
0
-3
-2
-1
0
1
2
3

Приклад 5.5

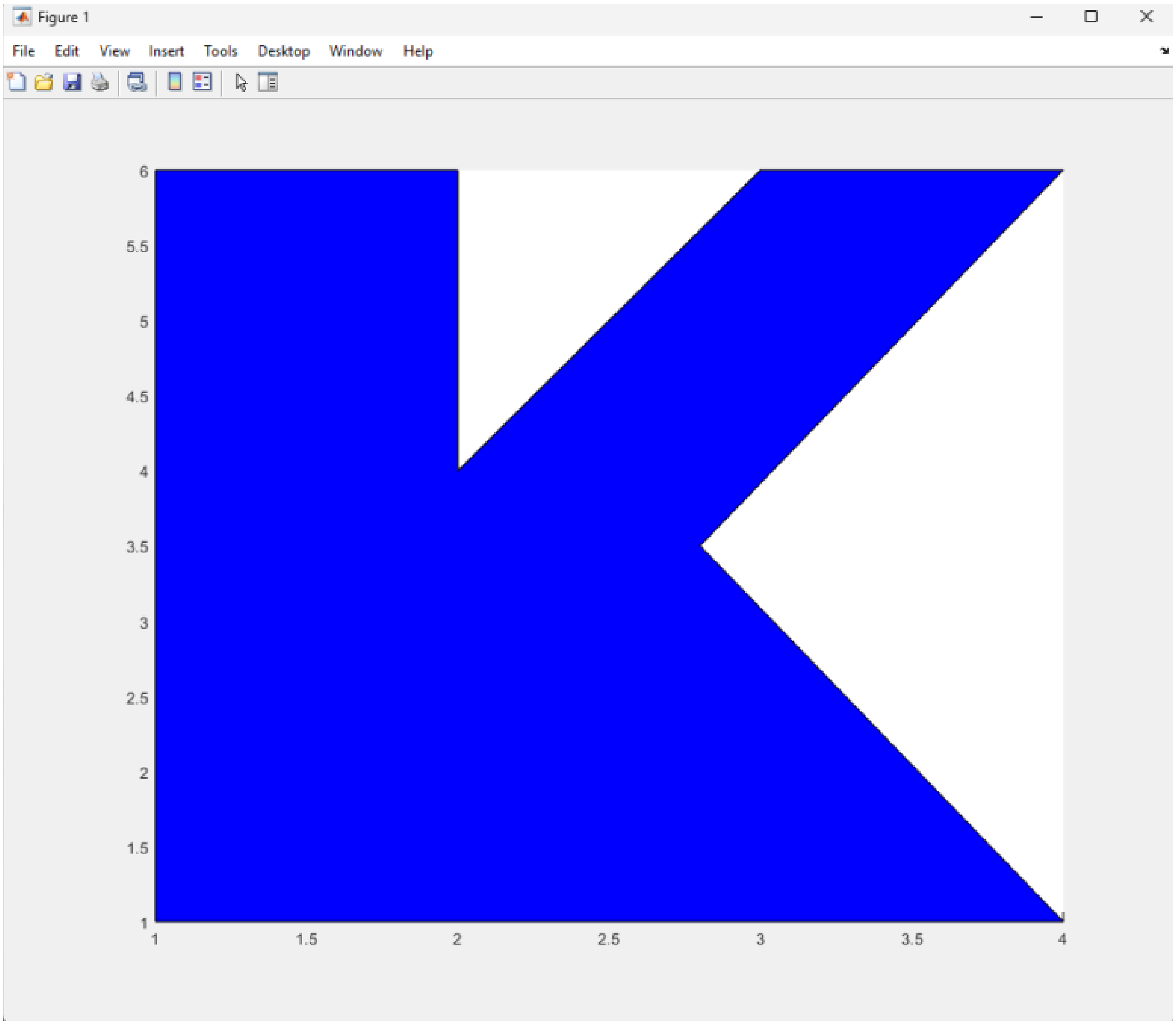
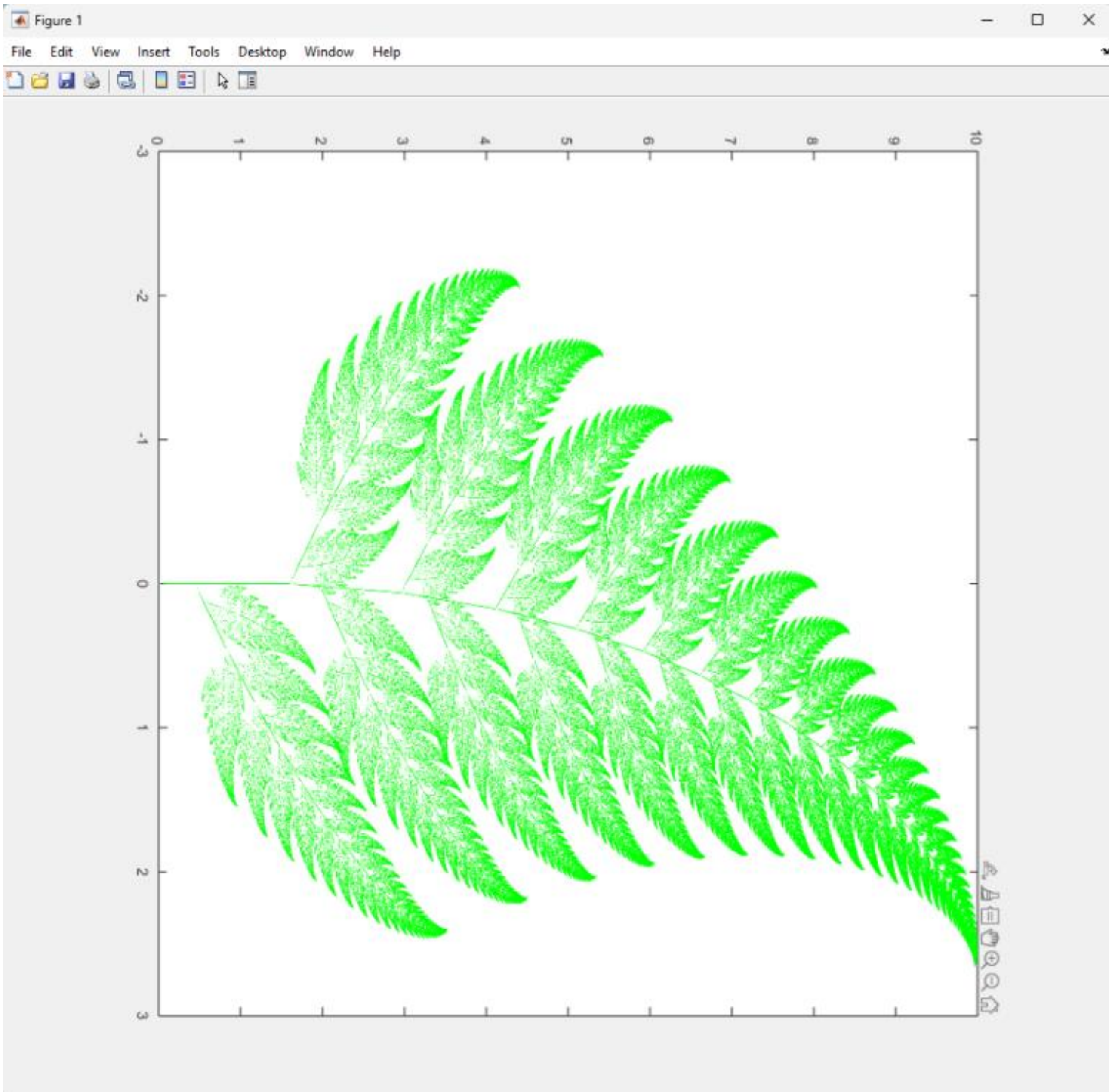


Figure 1
File Edit View Insert Tools Desktop Window Help
6
5.5
5
4.5
4
3.5
3
2.5
2
1.5
1
1 1.5 2 2.5 3 3.5 4

Приклад 5.6



Приклад 5.7

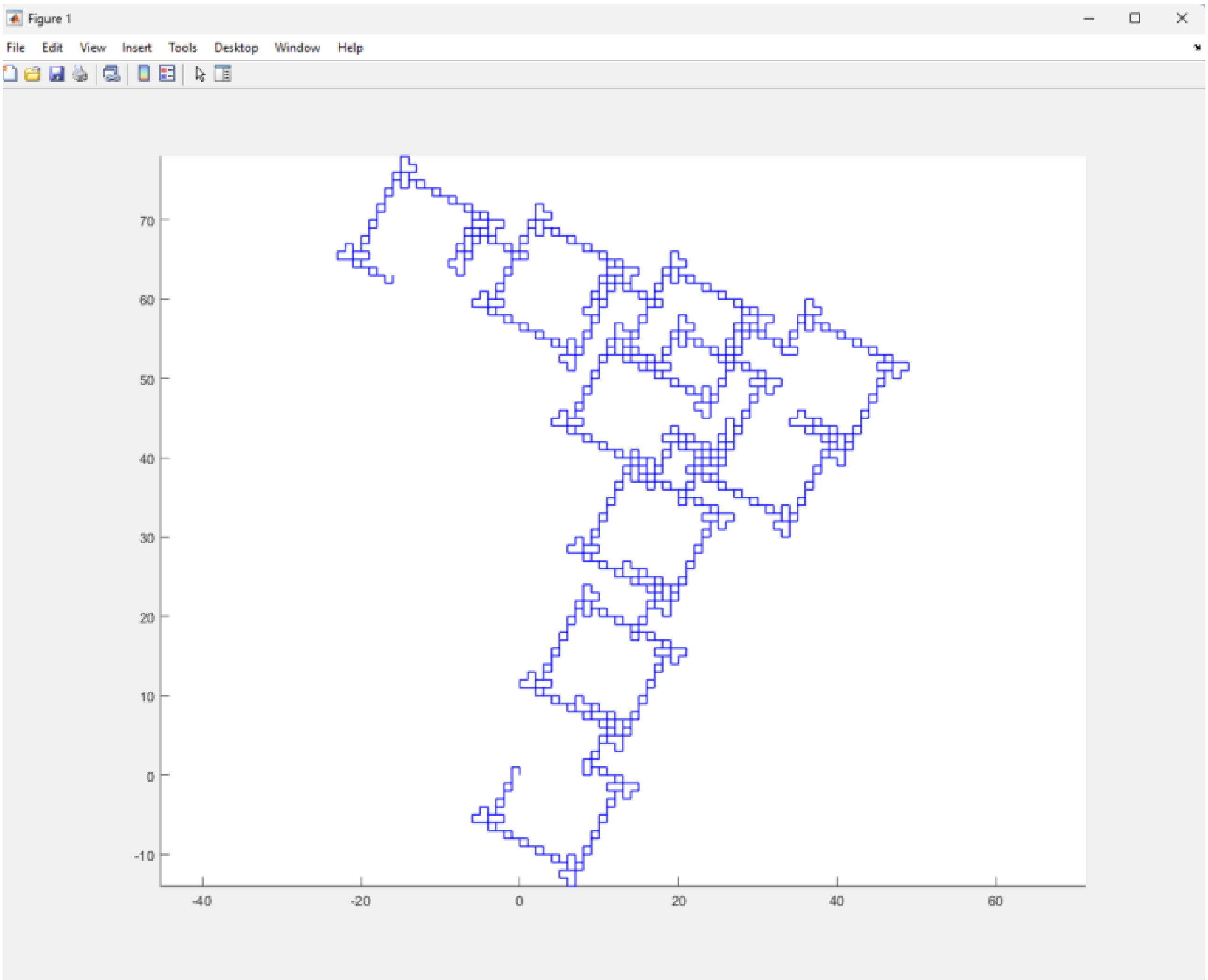


Figure 1
File Edit View Insert Tools Desktop Window Help
70
60
50
40
30
20
10
0
-10
-40
-20
0
20
40
60

Приклад 5.8

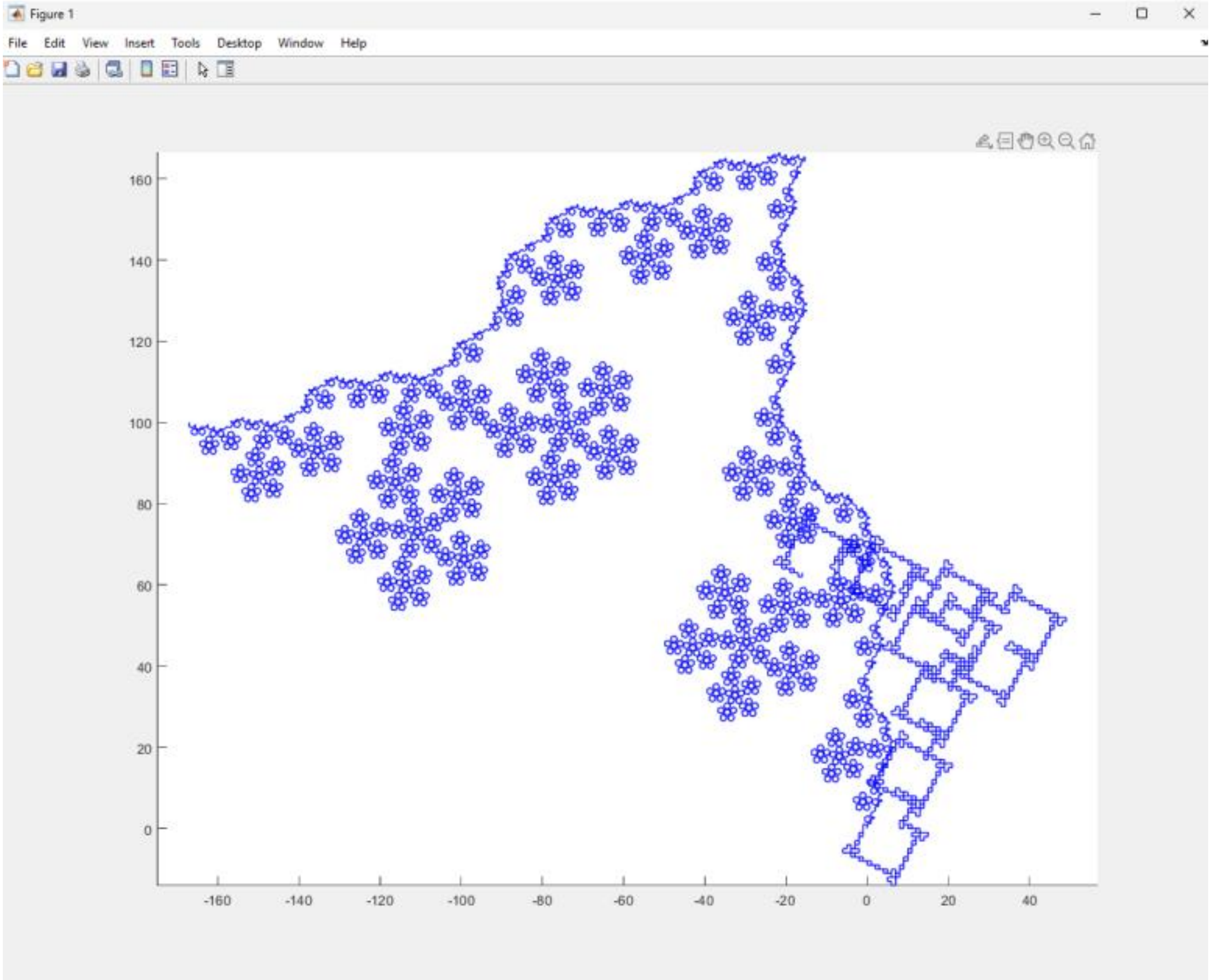


Figure 1
File Edit View Insert Tools Desktop Window Help
160
140
120
100
80
60
40
20
0
-160 -140 -120 -100 -80 -60 -40 -20 0 20 40

Приклад 5.9

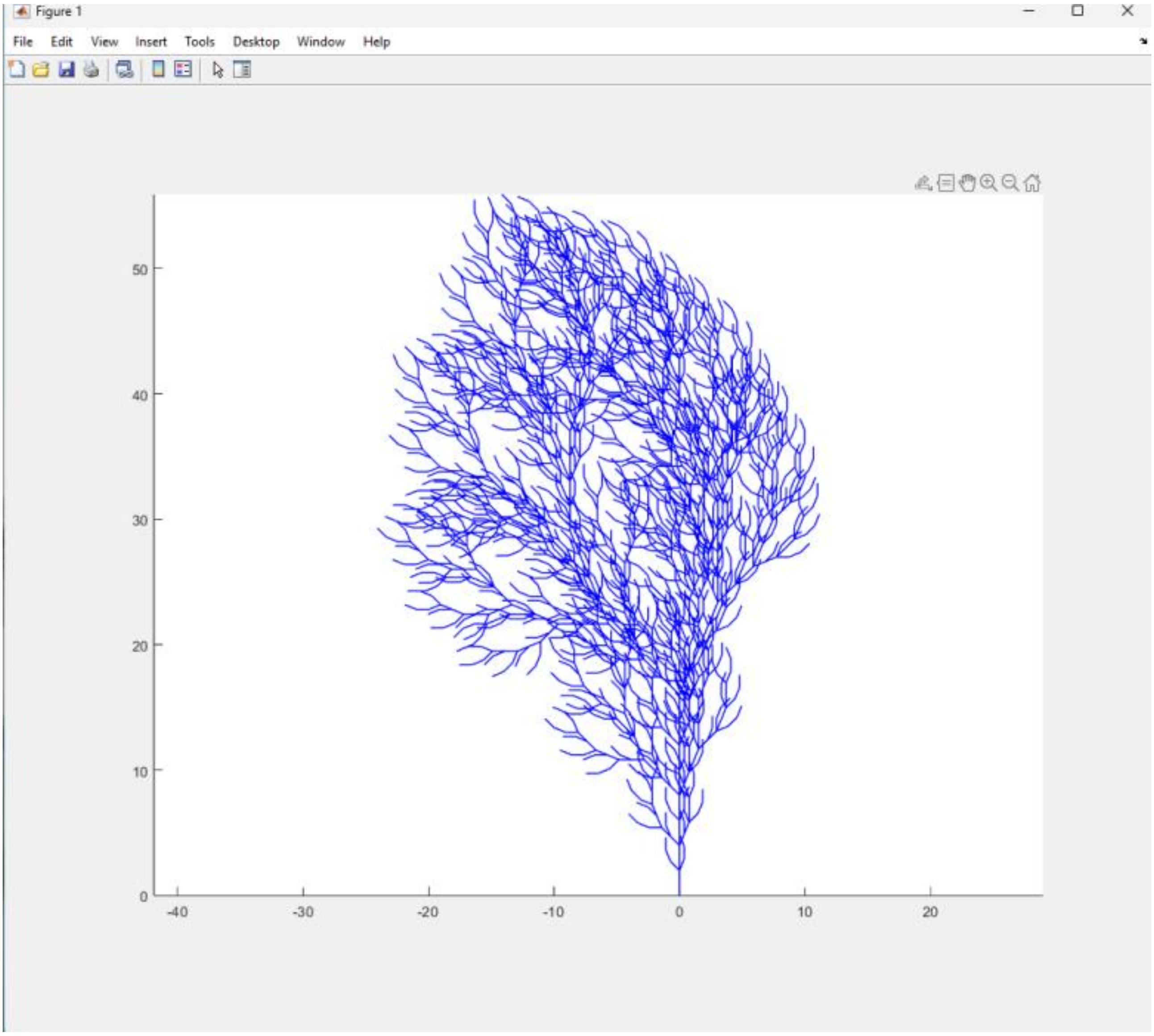


Figure 1
File Edit View Insert Tools Desktop Window Help
50
40
30
20
10
0
-40
-30
-20
-10
0
10
20

Приклад 5.10

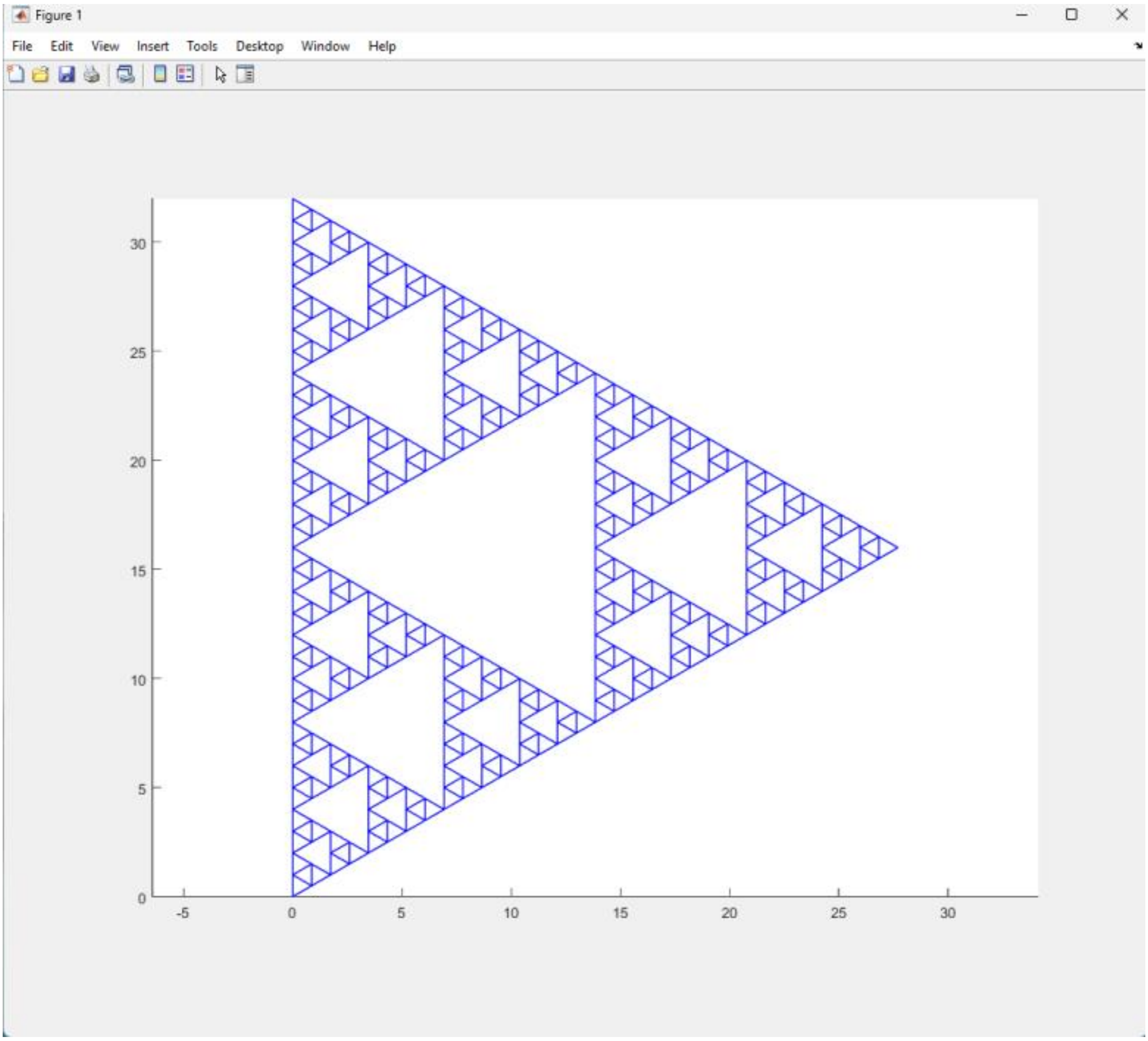


Figure 1
File Edit View Insert Tools Desktop Window Help
30
25
20
15
10
5
0
-5
0
5
10
15
20
25
30

Приклад 5.11

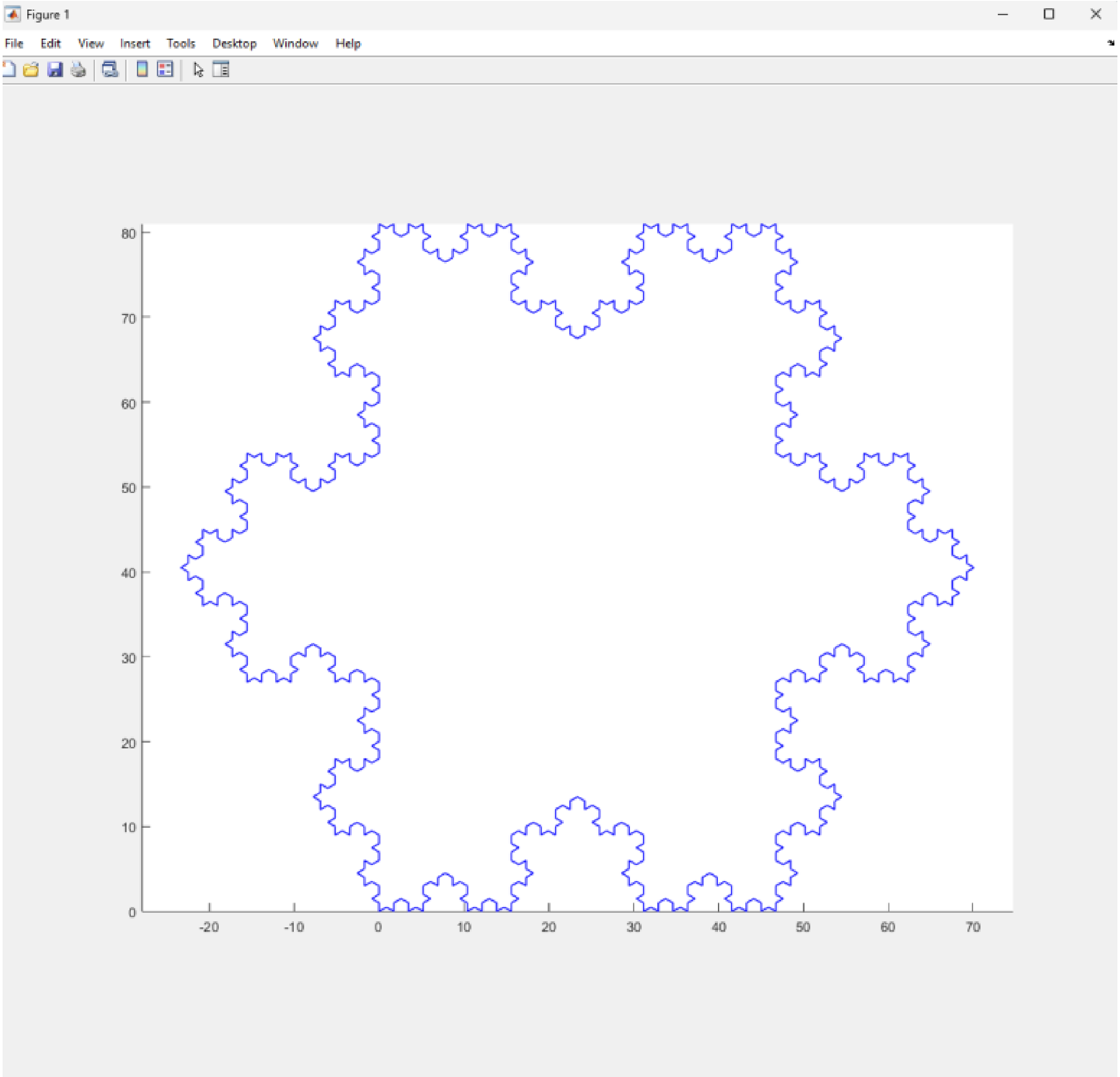


Figure 1
File Edit View Insert Tools Desktop Window Help
80
70
60
50
40
30
20
10
0
-20 -10 0 10 20 30 40 50 60 70

Приклад 5.12

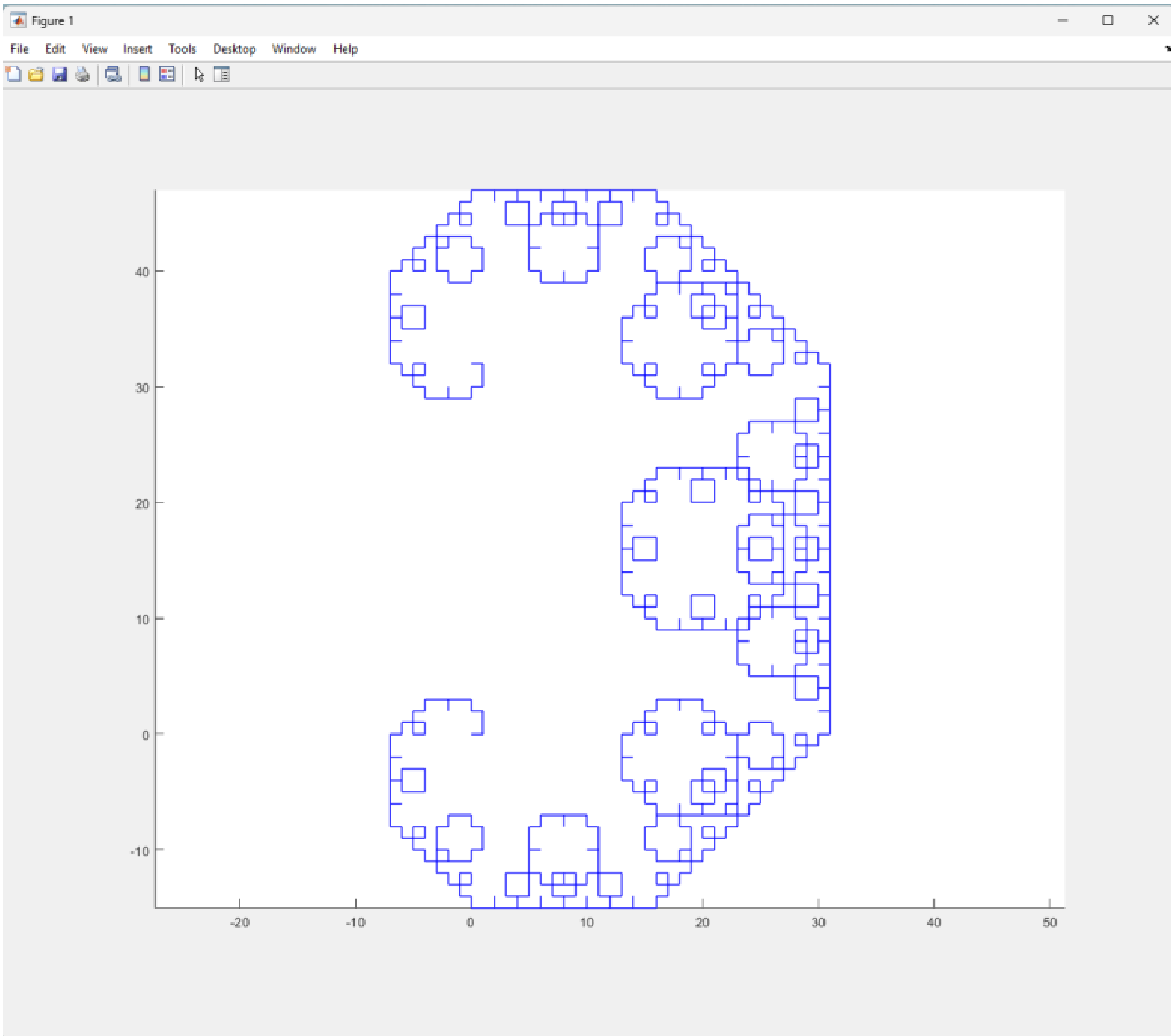


Figure 1
File Edit View Insert Tools Desktop Window Help
40
30
20
10
0
-10
-20
-10
0
10
20
30
40
50

Приклад 5.13

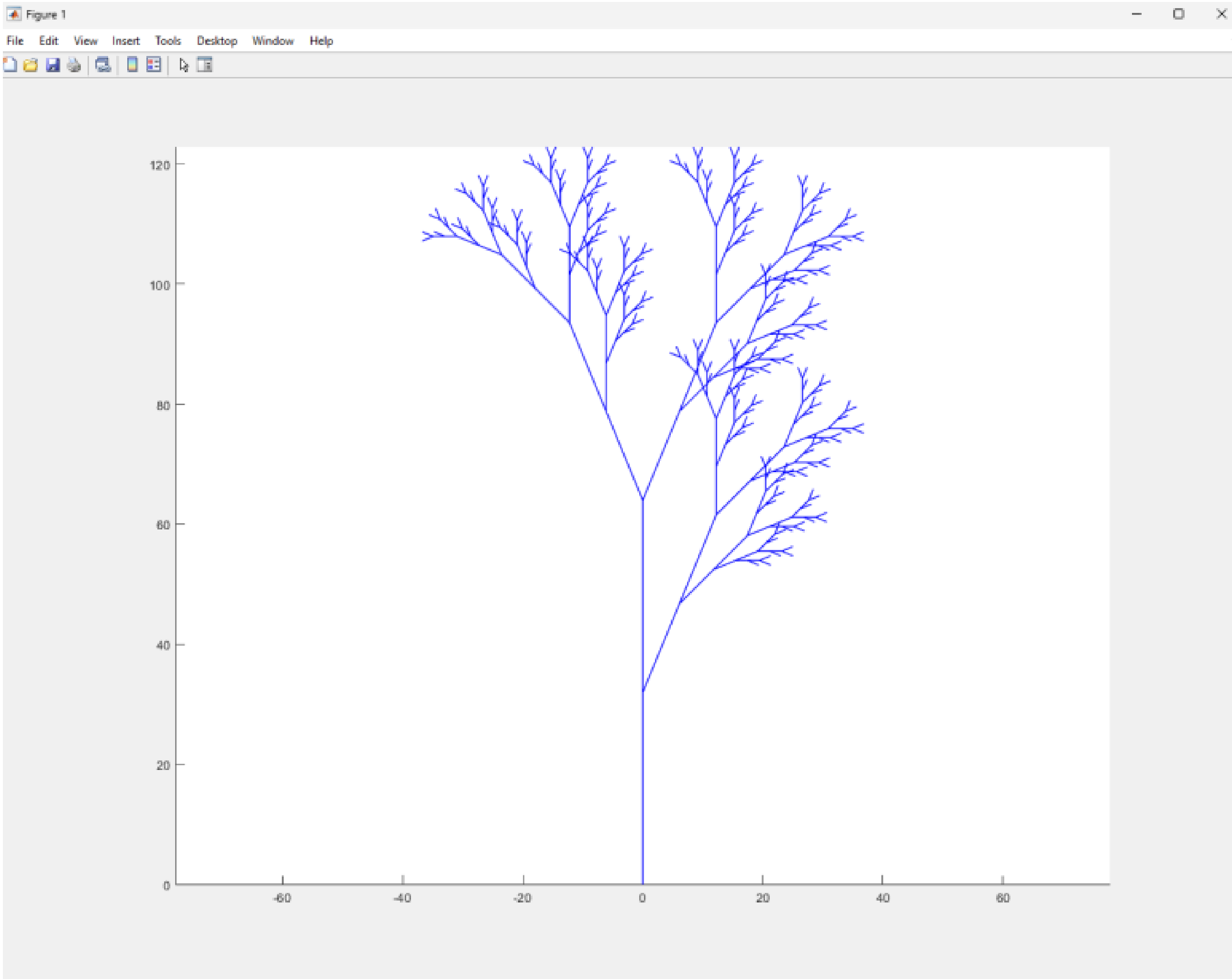


Figure 1
File Edit View Insert Tools Desktop Window Help
120
100
80
60
40
20
0
-60
-40
-20
0
20
40
60

Приклад 5.14

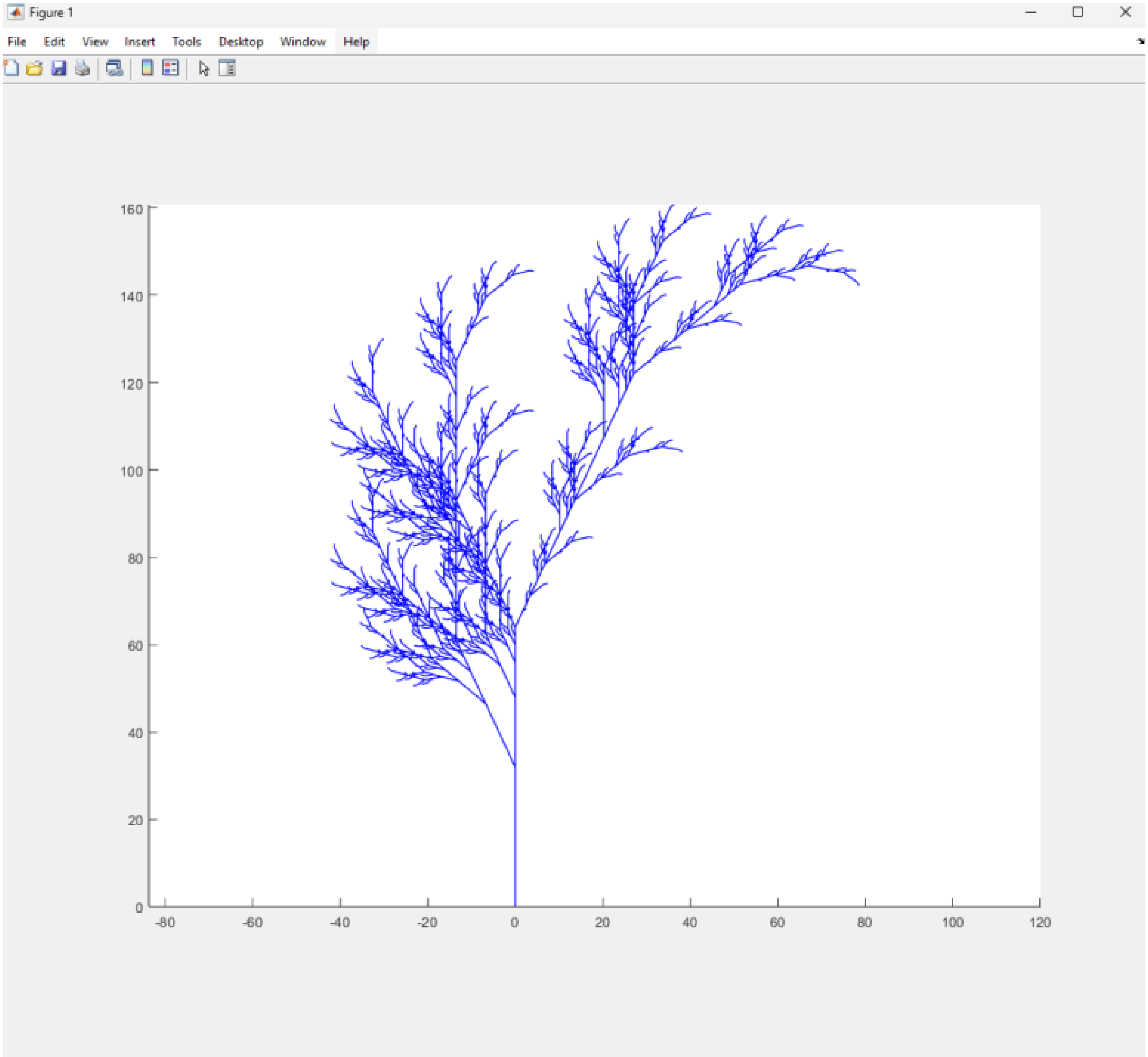


Figure 1
File Edit View Insert Tools Desktop Window Help
160
140
120
100
80
60
40
20
0
-80 -60 -40 -20 0 20 40 60 80 100 120

Приклад 5.15

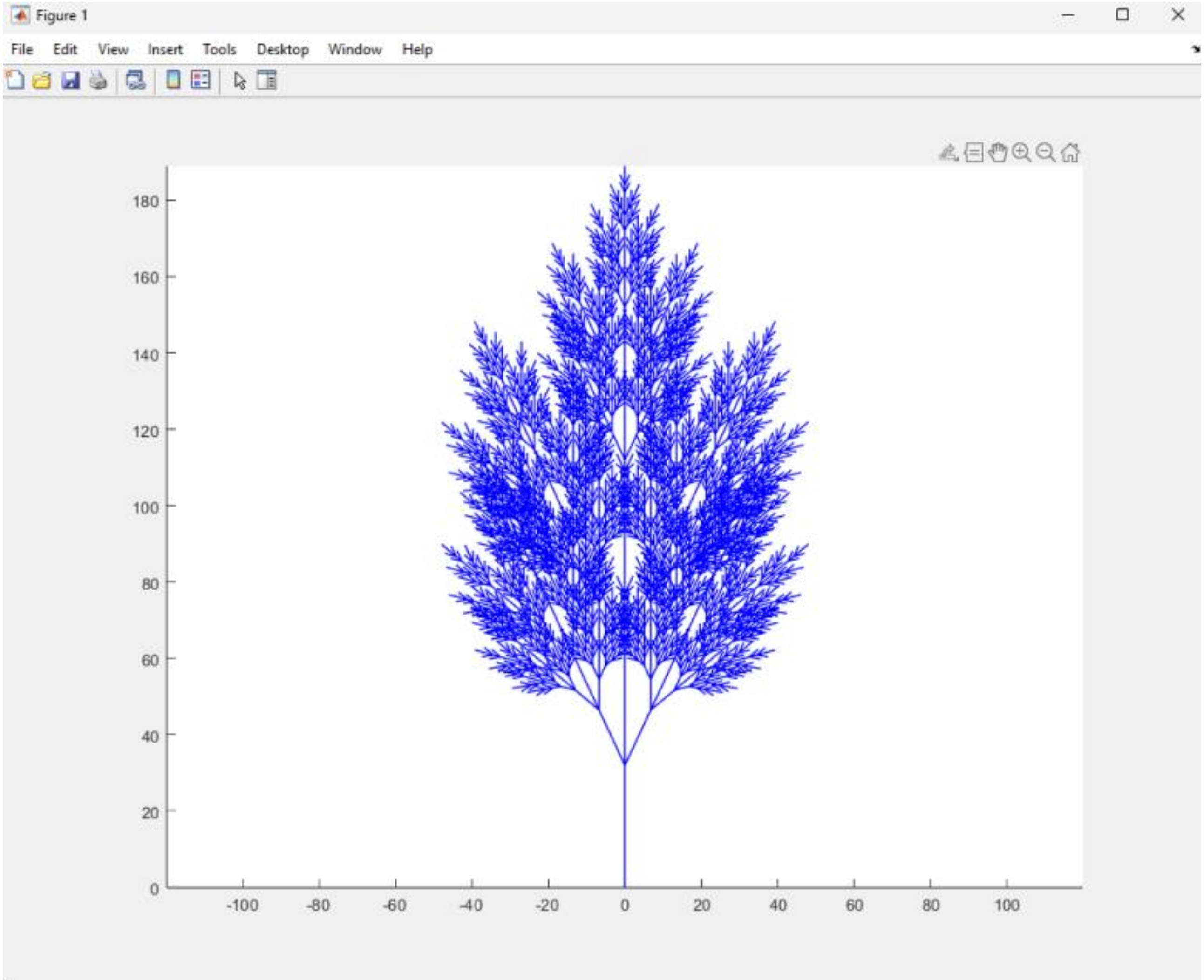


Figure 1
File Edit View Insert Tools Desktop Window Help
180
160
140
120
100
80
60
40
20
0
-100 -80 -60 -40 -20 0 20 40 60 80 100

Приклад 5.16

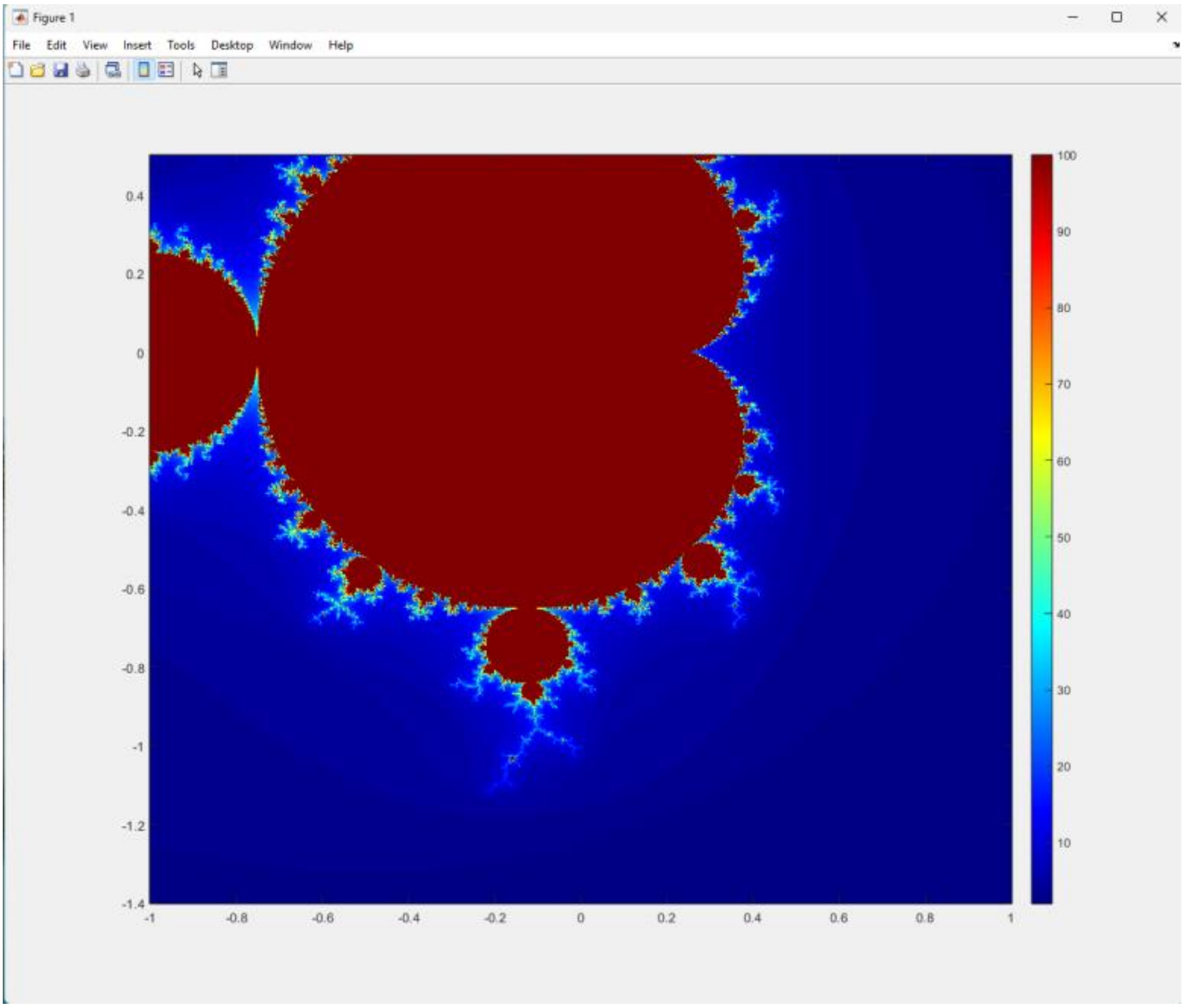


Figure 1
File Edit View Insert Tools Desktop Window Help
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1.2
-1.4
-1 -0.8 -0.6 -0.4 -0.2 0 0.2 0.4 0.6 0.8 1
100
90
80
70
60
50
40
30
20
10

Приклад 5.17

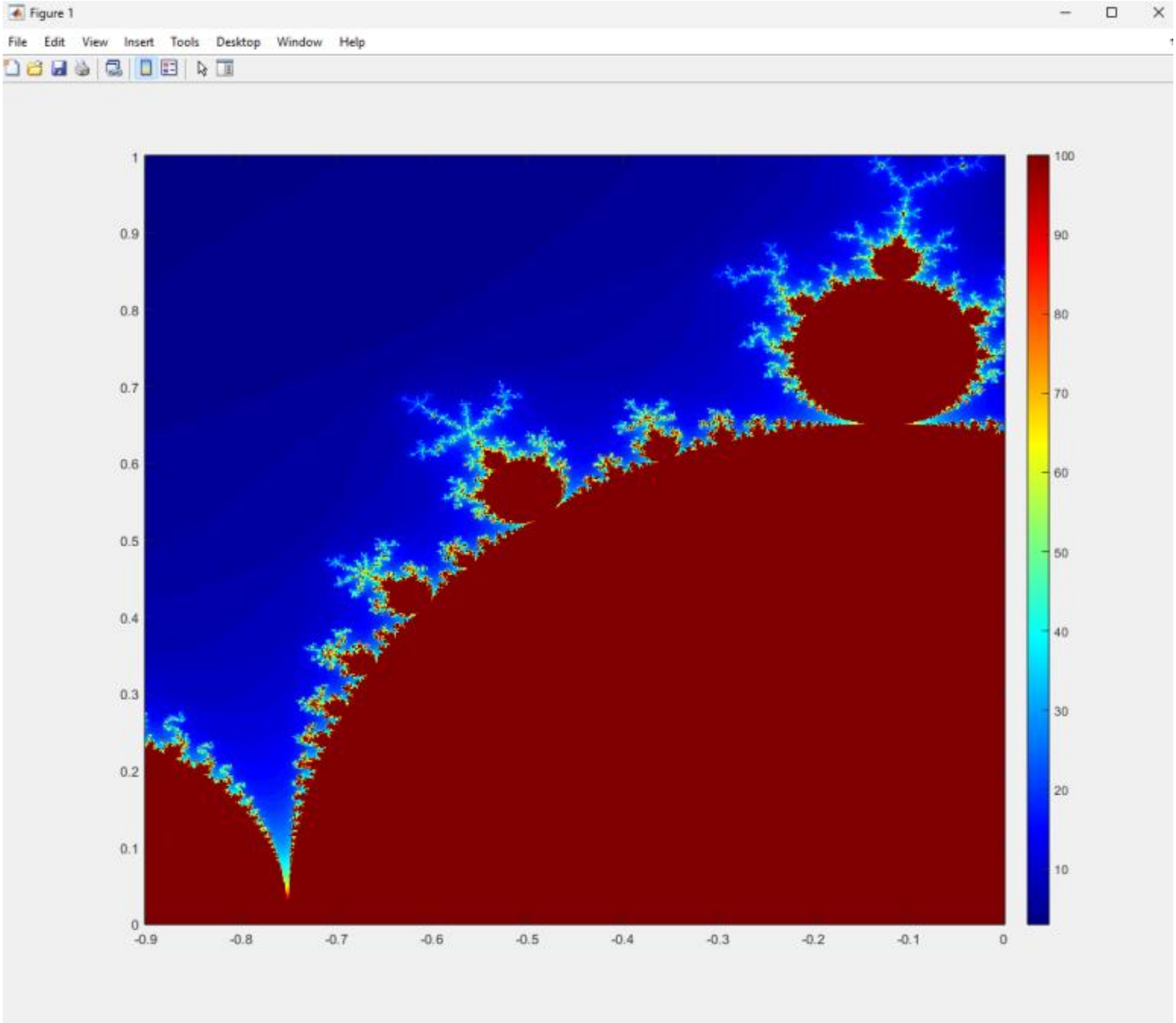


Figure 1
File Edit View Insert Tools Desktop Window Help
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.9 -0.8 -0.7 -0.6 -0.5 -0.4 -0.3 -0.2 -0.1 0
100
90
80
70
60
50
40
30
20
10

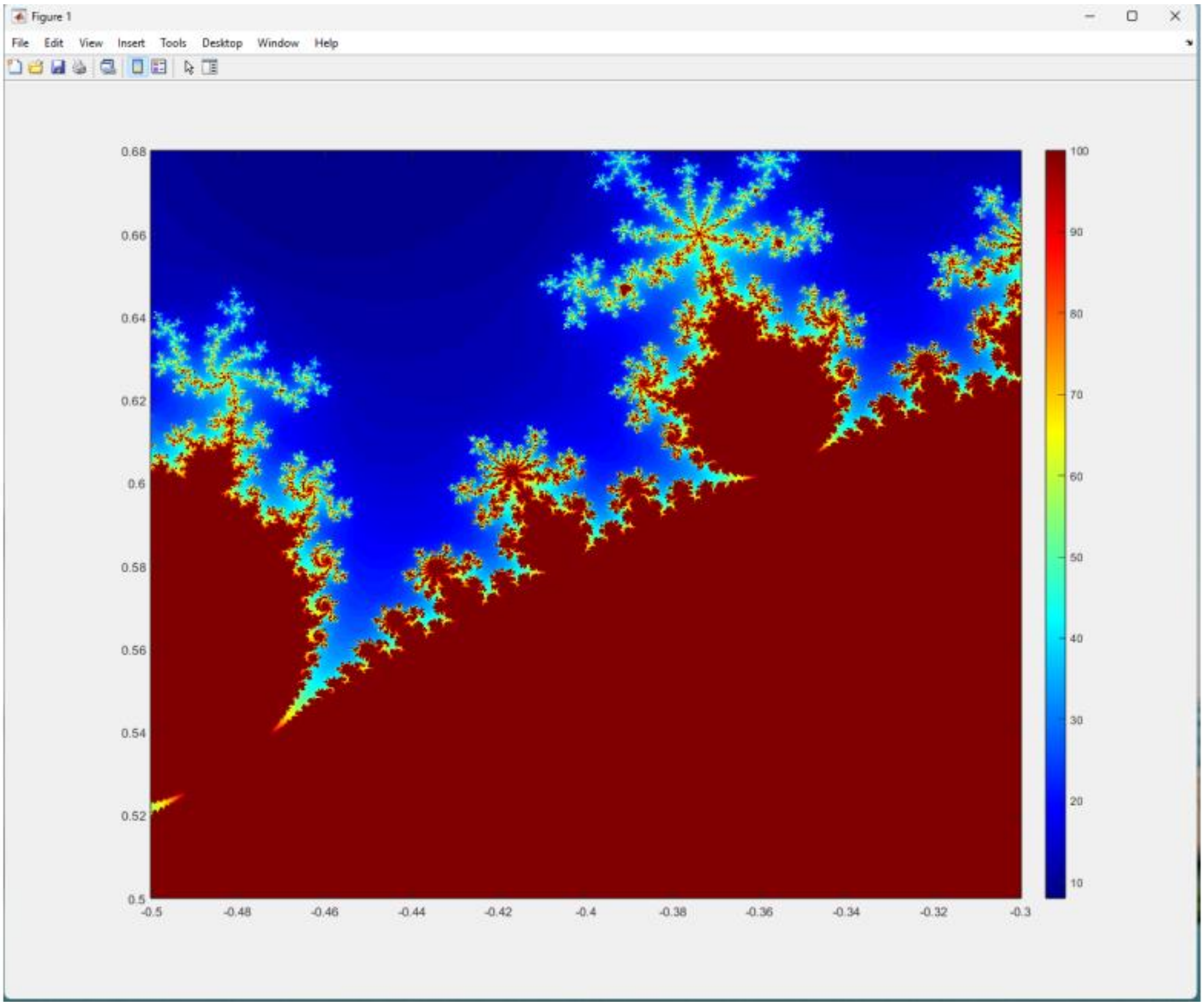


Figure 1
File Edit View Insert Tools Desktop Window Help
0.68
0.66
0.64
0.62
0.6
0.58
0.56
0.54
0.52
0.5
-0.5 -0.48 -0.46 -0.44 -0.42 -0.4 -0.38 -0.36 -0.34 -0.32 -0.3
100
90
80
70
60
50
40
30
20
10

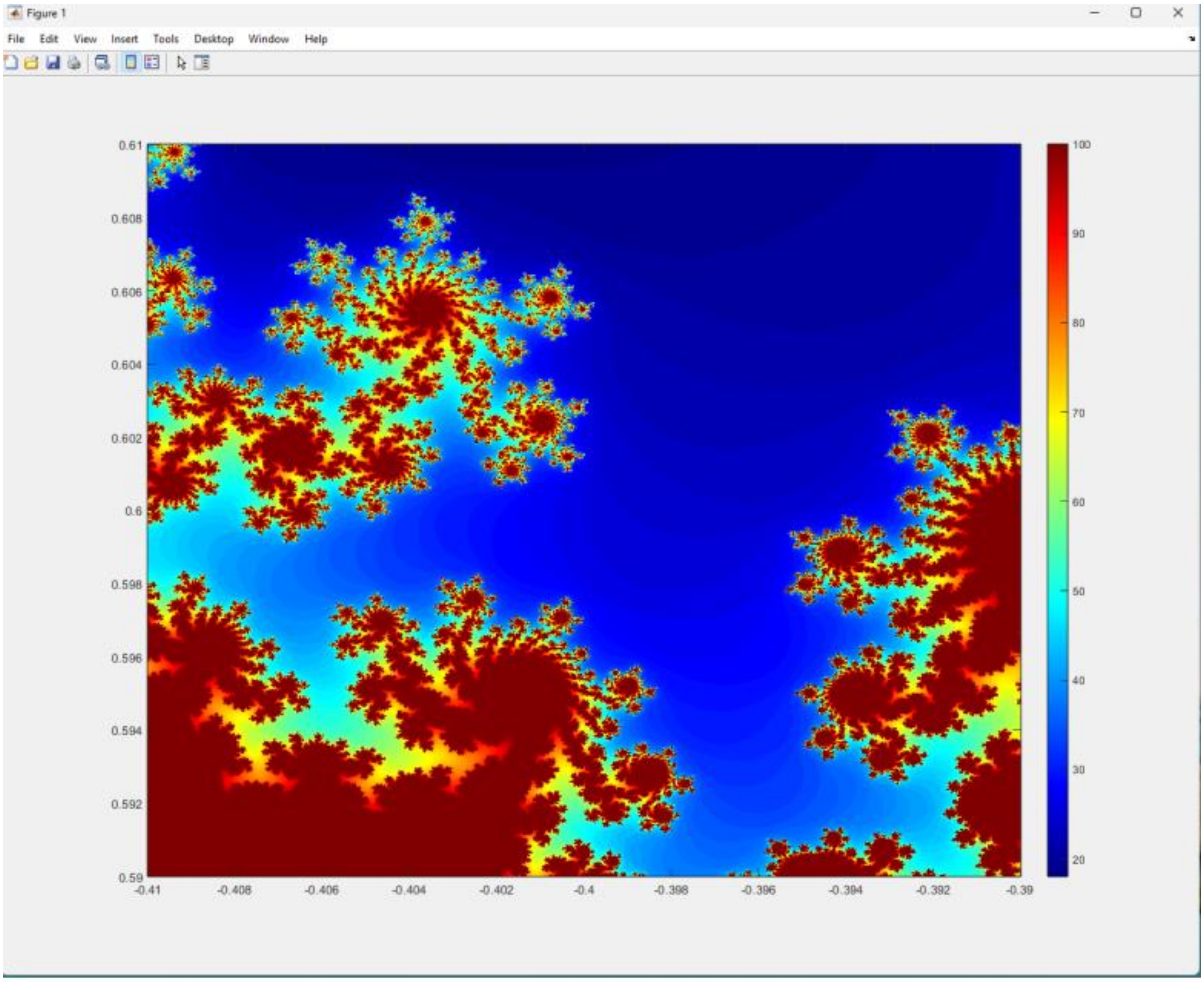
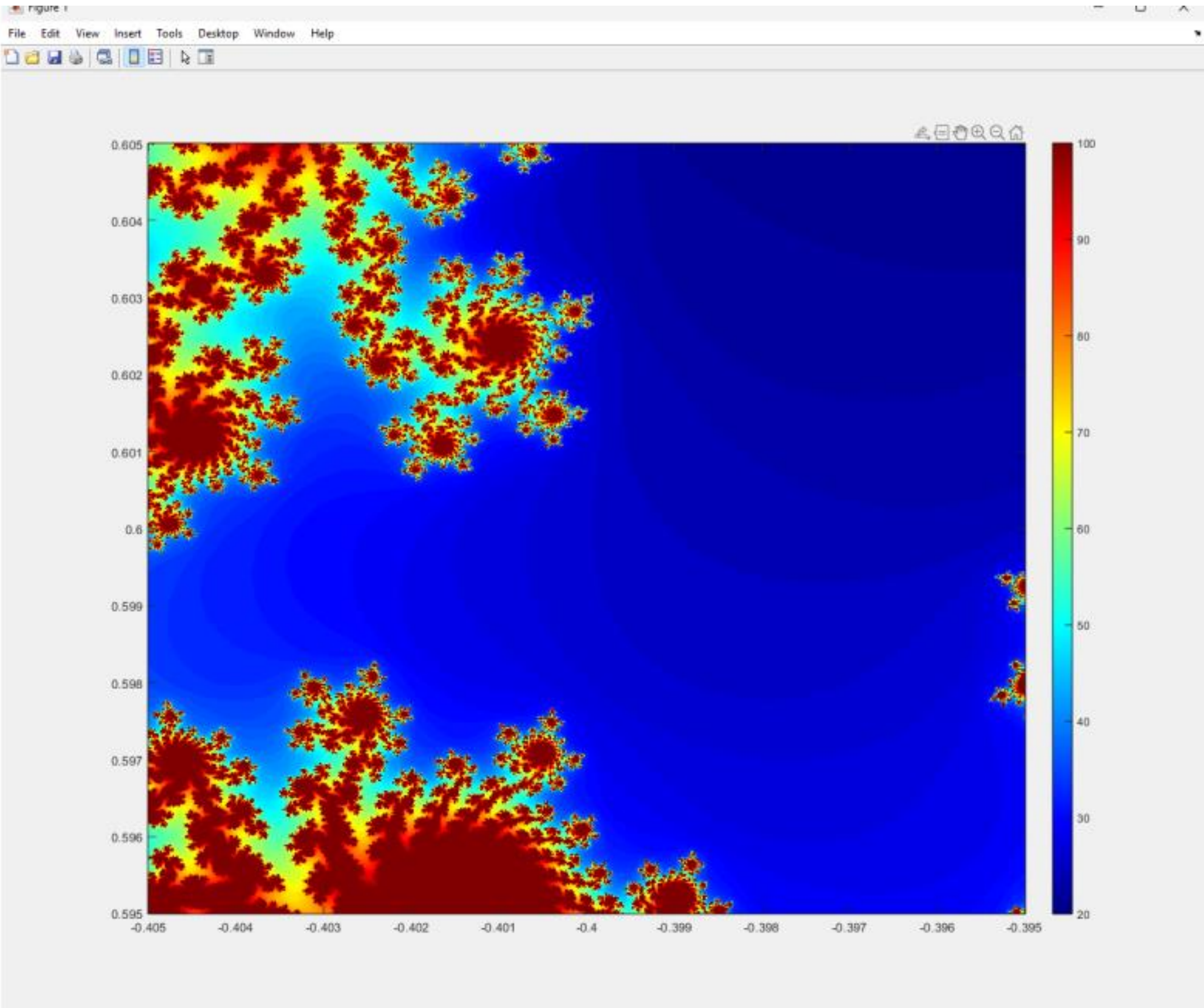
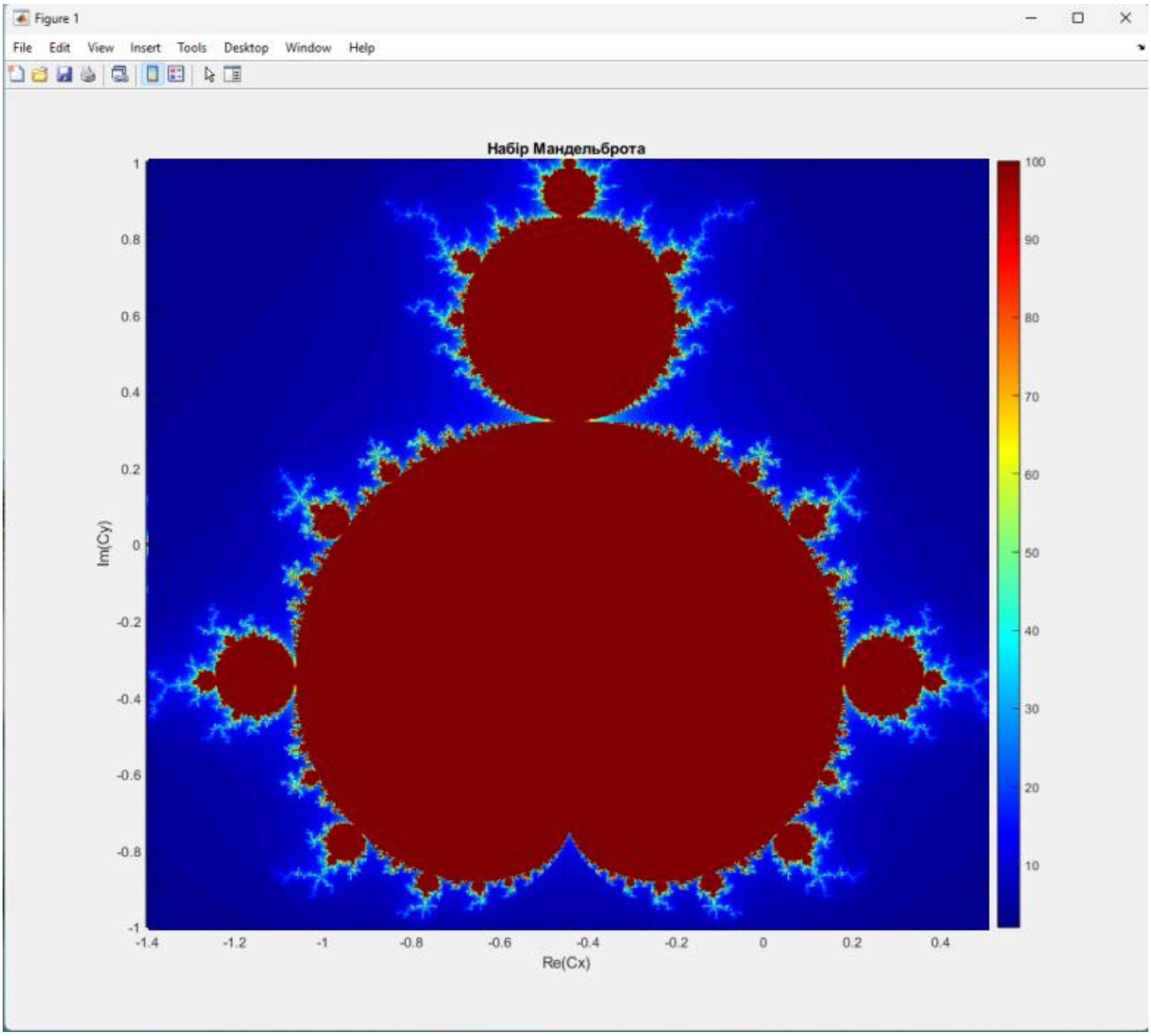


Figure 1
File Edit View Insert Tools Desktop Window Help
0.61
0.608
0.606
0.604
0.602
0.6
0.598
0.596
0.594
0.592
0.59
-0.41
-0.408
-0.406
-0.404
-0.402
-0.4
-0.398
-0.396
-0.394
-0.392
-0.39
100
90
80
70
60
50
40
30
20


File Edit View Insert Tools Desktop Window Help
0.605
0.604
0.603
0.602
0.601
0.6
0.599
0.598
0.597
0.596
0.595
-0.405
-0.404
-0.403
-0.402
-0.401
-0.4
-0.399
-0.398
-0.397
-0.396
-0.395
100
90
80
70
60
50
40
30
20

Приклад 5.18



Питання для самоконтролю

1. Як комп'ютер інтерпритує хаос?

Термін "хаос" у контексті комп'ютерних наук означає стан системи, який проявляється у непередбачуваному та виглядає хаотичним розподілі значень або станів. Хаос може виникнути у системах з чутливістю до початкових умов, де невеликі зміни можуть призвести до значущих відхилень у подальшому розвитку системи.

Існують методи, якими комп'ютери можуть інтерпретувати та моделювати хаотичні системи:

1. **Динамічні системи та рівняння:**

- Комп'ютери можуть використовувати рівняння для моделювання динамічних систем і вивчення їхньої поведінки. У хаотичних системах часто використовуються диференціальні рівняння чи різні різновиди дискретних рівнянь для опису станів системи в часі.

2. **Фрактали:**

- Фрактали - це геометричні об'єкти, які мають самоподібність на різних масштабах. Вони можуть бути використані для моделювання хаосу, оскільки їхня непередбачуваність та складність є характерними ознаками.

3. **Комп'ютерні програми для моделювання хаосу:**

- Існують спеціальні програми та мови програмування, призначені для моделювання та вивчення хаотичних систем. Наприклад, MATLAB, Python з бібліотеками, такими як NumPy та SciPy, можуть використовуватися для чисельного моделювання хаотичних систем.

4. **Випадкові числа:**

- Деякі аспекти хаосу можуть бути моделювані за допомогою випадкових чисел. Наприклад, випадкові форсування можуть бути введені у рівняння для створення хаотичних змін.

5. **Теорія хаосу та аттрактори:**

- Теорія хаосу надає математичні інструменти для аналізу та розуміння хаотичних систем. Поняття аттракторів та фазових портретів використовуються для опису динаміки хаотичних систем.

Важливо відзначити, що хаос не означає абсолютної випадковості, а лише позначає дуже чутливу до початкових умов та непередбачувану динаміку системи. Комп'ютери дозволяють вивчати та розуміти цю непередбачуваність у хаотичних системах за допомогою моделювання та чисельних методів.

2. Якою послідовністю дій реалізується побудова листка папороті?

Побудова листка папороті відбувається за допомогою алгоритму, відомого як алгоритм папороті (Fern Fractal). Цей алгоритм використовує випадковість для генерації структури, схожої на форму листя папороті. Основні етапи цього алгоритму можна описати наступним чином:

1. **Ініціалізація:**

- Визначте початкову точку (x, y) на площині.

2. **Ітерації:**

- Застосовуйте серію перетворень до кожної точки для генерації нових точок. Кожне перетворення асоційоване з певною ймовірністю. Ці ймовірності можуть бути налаштовані так, щоб вони відображали особливості форми папороті.

3. **Перетворення:**

- Зазвичай використовуються три основних види перетворень:

- **Перетворення зміщення:**

$$x_{n+1} = 0 \cdot x_n + 0 \cdot y_n + 0$$
$$y_{n+1} = 0 \cdot x_n + 0.16 \cdot y_n$$

\[x_{n+1} = 0 \cdot x_n + 0 \cdot y_n + 0 \]

\[y_{n+1} = 0 \cdot x_n + 0.16 \cdot y_n \]

- **Перетворення зміщення та масштабування:**

$$x_{n+1} = 0.85 \cdot x_n + 0.04 \cdot y_n$$
$$y_{n+1} = -0.04 \cdot x_n + 0.85 \cdot y_n + 1.6$$

\[x_{n+1} = 0.85 \cdot x_n + 0.04 \cdot y_n \]

\[y_{n+1} = -0.04 \cdot x_n + 0.85 \cdot y_n + 1.6 \]

- **Перетворення зміщення, масштабування та обертання:**

$$x_{n+1} = 0.2 \cdot x_n - 0.26 \cdot y_n$$
$$y_{n+1} = 0.23 \cdot x_n + 0.22 \cdot y_n + 1.6$$

\[x_{n+1} = 0.2 \cdot x_n - 0.26 \cdot y_n \]

\[y_{n+1} = 0.23 \cdot x_n + 0.22 \cdot y_n + 1.6 \]

- Кожне з цих перетворень застосовується з певною ймовірністю. Наприклад, з ймовірністю 1% може застосовуватися перетворення зміщення, а з ймовірністю 85% - перетворення зміщення та масштабування.

4. **Повторення:**

- Повторюйте цей процес для кожної нової точки, побудованої на попередньому етапі.

5. **Відображення:**

- Відобразіть отримані точки на графіку.

Цей алгоритм є прекрасним прикладом використання випадковості та деталей математичних перетворень для створення складної та красивої графіки.

3. Наведіть геометричну інтерпретацію геометричних перетворень на площині.

Геометричні перетворення на площині змінюють положення та форму геометричних об'єктів. Ось кілька основних геометричних перетворень та їх геометрична інтерпретація:

1. **Трансляція (зміщення):**

- **Операція:**

\[T(x, y) = (x + a, y + b) \]

$$T(x, y) = (x + a, y + b)$$

- **Геометрична інтерпретація:**

Трансляція зміщує кожен піксель чи точку на площині на фіксовану величину \(a\) уздовж вісі \(x\) і \(b\) уздовж вісі \(y\). Це еквівалентно переміщенню об'єкта на площині без зміни його розмірів чи форми.

2. **Масштабування:**

- **Операція:**

$$S(x, y) = (k \cdot x, k \cdot y)$$

\[S(x, y) = (k \cdot x, k \cdot y) \]

- **Геометрична інтерпретація:**

Масштабування змінює розмір об'єкта на площині. Значення \(k > 1\) збільшує розміри, а \(0 < k < 1\) зменшує розміри об'єкта вздовж обох вісей. Це еквівалентно розтягуванню чи стисненню об'єкта.

3. **Обертання:**

- **Операція:**

$$R(x, y, \theta) = (x \cdot \cos(\theta) - y \cdot \sin(\theta), x \cdot \sin(\theta) + y \cdot \cos(\theta))$$

\[R(x, y, \theta) = (x \cdot \cos(\theta) - y \cdot \sin(\theta), x \cdot \sin(\theta) + y \cdot \cos(\theta)) \]

- **Геометрична інтерпретація:**

Обертання обертає об'єкт навколо початку координат на кут \(\theta\) уздовж годинникової стрілки. Змінюючи значення \(\theta\), можна отримати обертання навколо інших точок чи в інших напрямках.

4. **Відображення (симетрія відносно вісі):**

- **Операція:**

$$F(x, y) = (x, -y) \text{ або } F(x, y) = (-x, y)$$

\[F(x, y) = (x, -y) \] або \[F(x, y) = (-x, y) \]

- **Геометрична інтерпретація:**

Відображення відносно вісі \(y\) (\(F(x, y) = (x, -y)\)) створює симетричне відображення об'єкта відносно вісі \(y\). Так само, відображення відносно вісі \(x\) (\(F(x, y) = (-x, y)\)) створює симетричне відображення об'єкта відносно вісі \(x\).

Ці базові геометричні перетворення є важливими для обробки та моделювання графічних об'єктів на площині та у комп'ютерній графіці.

4. Що таке L-система, яка породжує правило?

L-система (або система Лінденмайера) є математичною системою, яка використовується для моделювання росту рослин, гілок, архітектурних структур та інших природних форм. Вона була запропонована математиками Аристідом Лінденмайером та Петером Прусінські в 1968 році. L-система включає в себе два основних компонента: початковий набір символів (аксіома) і набір правил, які визначають, як кожен символ повинен бути замінений на інший символ чи послідовність символів.

Основні елементи L-системи:

1. **Аксіома (початковий набір символів):**

- Це початковий рядок символів, який визначає початковий стан системи.

2. **Правила перетворення (продукційні правила):**

- Це набір правил, які визначають, які символи повинні бути замінені на інші символи на кожному кроці ітерації. Кожне правило складається з символу для заміни та послідовності символів, якою його слід замінити.

Процес використання L-системи для генерації структури можна описати наступним чином:

1. **Початок:**

- Визначте аксіому - початковий набір символів.

2. **Ітерація:**

- Застосуйте правила перетворення до кожного символу в аксіомі, щоб сформувати новий рядок символів. Повторюйте цей процес для кількох ітерацій.

3. **Візуалізація:**

- Присвойте кожному символу певні геометричні або дійсні характеристики (наприклад, кут обертання, довжина, кольорова палітра тощо) і візуалізуйте отриманий результат.

Приклад L-системи для генерації фрактального дерева (використовуючи символи "F" для переміщення вперед, "+" та "-" для обертань) може виглядати наступним чином:

- Аксіома: F

- Правила:

- F -> FF+[+F-F-F]-[-F+F+F]

Це лише один приклад, і L-системи можуть використовуватися для створення різноманітних геометричних структур у природі та мистецтві.

5. Якою послідовністю дій реалізується побудова кривої "Кущ"?

Крива "Кущ" (Bush) є геометричною фігурою, яку можна згенерувати за допомогою системи Лінденмайера (L-системи). Давайте розглянемо приклад, як можна побудувати криву "Кущ" за допомогою L-системи.

Аксіома (початковий набір символів):

\[F \]

Правила перетворення (продукційні правила):

\[F \rightarrow FF+[+F-F-F]-[-F+F+F] \]

Тут символи в правилах мають такі значення:

- \(F \) вказує на переміщення вперед на фіксовану відстань і зображає гілку.

- \(+ \) та \(- \) вказують на обертання вправо та вліво відповідно.

- \([\) та \( ] \) вказують на збереження та відновлення поточного стану для створення гілкової структури.

Початково ви маєте один символ \(F \), і на кожній ітерації кожен символ замінюється відповідно до правил. Повторення цього процесу створює складену гілкову структуру, схожу на гілки куща чи дерева.

Ітерація 1:

$$F \xrightarrow{\text{правило}} FF+[+F-F-F]-[-F+F+F]$$

Ітерація 2 (заміна кожного \(F \) у попередньому рядку):

$$FF+[+F-F-F]-[-F+F+F] \xrightarrow{\text{правило}} FFFF+[+FF-F-F]-[-F+F+F]+[+FF-F-F]-[-F+F+F]-FF+[+F-F-F]-[-F+F+F]$$

І так далі.

Цей процес можна повторювати стільки разів, скільки потрібно для отримання бажаного вигляду кривої "Кущ". Визначені кути обертання та відстані переміщення можуть бути адаптовані для отримання різних форм і розмірів гілок.

6. Якою послідовністю дій реалізується побудова трикутника Серпінського?

Трикутник Серпінського - це геометрична структура, яку можна побудувати за допомогою рекурсивного процесу ділення трикутника на менші трикутники. Для побудови трикутника Серпінського зазвичай використовуються рекурсивні методи. Основна ідея полягає в тому, щоб розділити трикутник на три менших трикутники і рекурсивно застосовувати цей процес до кожного з менших трикутників.

Почнемо з початкового трикутника. Позначимо його вершини як A, B і C.

1. **Крок 1 (Базовий випадок):**

- Нам потрібно побудувати трикутник Серпінського для початкового трикутника ABC.

2. **Крок 2 (Рекурсія):**

- Розділіть трикутник на три менших трикутники, наприклад, за допомогою серединних точок сторін трикутника. Позначте ці точки як M1, M2 і M3.

- Застосуйте той самий процес до кожного з отриманих менших трикутників. Рекурсивно викликайте процес для трикутників (A, M1, M2), (M1, B, M3) і (M2, M3, C).

Виконуючи ці кроки рекурсивно для кожного нового трикутника, ви побачите, що на кожному рівні рекурсії буде додано нові трикутники, утворюючи структуру трикутника Серпінського.

Програматично це може бути втілено за допомогою мов програмування та графічних бібліотек для відображення графіки. В багатьох мовах програмування, таких як Python з бібліотекою turtle чи JavaScript з використанням HTML і canvas, ви можете написати код для рекурсивної побудови трикутника Серпінського.

7. Чим відрізняється аксіоми і породжуючі правила, що використовуються під час побудови сніжинки Коха і "фрактальних" рослин?

Аксіоми та породжуючі правила є основними елементами систем Лінденмайера (L-систем), які використовуються для генерації фрактальних зображень, таких як сніжинка Коха та фрактальні рослини. Давайте розглянемо їх відмінності:

1. **Аксіома (початковий набір символів):**

- **Сніжинка Коха:** Аксіома це одна чи декілька початкових символів, з яких починається конструкція. У випадку сніжинки Коха аксіома може бути одним відрізком (символ F), і з цього відрізка буде виходити сама сніжинка за допомогою правил.

- **Фрактальні рослини:** Аксіома для фрактальних рослин може представляти початкову форму чи структуру рослини.

2. **Породжуючі правила (продукційні правила):**

- **Сніжинка Коха:** Породжуючі правила визначають, як символи розглядаються та замінюються на кожній ітерації. У випадку сніжинки Коха, одне з основних правил може мати вигляд:

$$F \rightarrow F - F + + F - F$$

\[F \rightarrow F-F++F-F \]

Це означає, що кожен символ F буде замінений на послідовність F-F++F-F.

- **Фрактальні рослини:** Породжуючі правила для фрактальних рослин визначають, як виглядає кожна частина рослини та як вона розгалужується. Наприклад:

$$F \rightarrow F + F - F - F + F$$

\[F \rightarrow F+F-F-F+F \]

Це може означати, що кожен символ F буде замінений на послідовність F+F-F-F+F.

Отже, аксіома визначає початковий стан системи, а породжуючі правила вказують, як цей стан буде змінюватися на кожному кроці рекурсії чи ітерації. Обидва ці елементи разом дозволяють конструювати складні та красиві фрактальні зображення за допомогою систем Лінденмайера.

8. Покажіть подібність та відмінності під час побудови листка папороті та фрактала Мальденборта.

Побудова листка папороті та фрактала Мальденборта використовує концепції систем Лінденмайера (L-систем). Давайте порівняємо їх подібності та відмінності:

1. **Подібності:**

- **Використання L-систем:**

Обидва фрактали використовують L-системи для їхньої побудови. L-система визначає початковий набір символів (аксіому) та правила перетворення символів на кожній ітерації.

- **Рекурсивний процес:**

Обидва фрактали будуються за допомогою рекурсивного процесу, в якому на кожному кроці розгалужується кожна одиниця структури.

- **Геометричні деталі:**

Обидва фрактали намагаються моделювати складні геометричні структури з допомогою простих правил та рекурсії.

2. **Відмінності:**

- **Призначення та форма:**

- *Листок папороті:* Зазвичай спрямований на моделювання форми листя папороті, що має властивості самоподібності.

- *Фрактал Мальденборта:* Це більш абстрактний фрактал, який часто використовується в графіці та мистецтві, і не обов'язково моделює конкретні природні об'єкти.

- **Породжуючі правила та конструкція:**

- *Листок папороті:* Використовує специфічні породжуючі правила, щоб моделювати гілля та листя папороті.

- *Фрактал Мальденборта:* Має свої унікальні правила та конструкції, які роблять його відмінним від інших фракталів.

- **Варіації та параметри:**

- *Листок папороті:* Зазвичай використовується з декількома варіаціями параметрів для створення різних форм папороті.

- *Фрактал Мальденборта:* Може бути змінений та варійований за допомогою різних параметрів, що впливають на його вигляд та структуру.

У обох випадках використання L-систем дозволяє створювати цікаві та складні геометричні фігури за допомогою відносно простих правил та рекурсії.

9. Як називається початкова послідовність символів?

Початкова послідовність символів в системі Лінденмайера називається **аксіомою**. Аксіома є початковим набором символів або символьною стрічкою, з якої розпочинається конструкція за допомогою правил перетворення на кожній ітерації чи кроці рекурсії. У контексті L-систем аксіома визначає початковий стан системи перед будь-якими перетвореннями.

10. Назвіть характерну особливість фрактала.

Характерна особливість фрактала - це **самоподібність**. Самоподібність означає те, що фрактал має структуру, яка повторюється на різних масштабах. Це означає, що частини фрактала подібні за формою до всього фрактала. Навіть, якщо збільшити (або зменшити) частину фрактала, вона залишиться подібною до вихідного фрактала.

Це властивість робить фрактали відмінними від традиційних геометричних фігур, оскільки вони можуть мати складні та деталізовані структури на будь-якому масштабі. Самоподібність є ключовим елементом багатьох фракталів і робить їх корисними для моделювання природних об'єктів та явищ, таких як хмари, гори, дерева та інше, оскільки ця властивість часто властива природним формам.