Лабораторна робота №1

Студента групи КН-11

Сеня Тараса

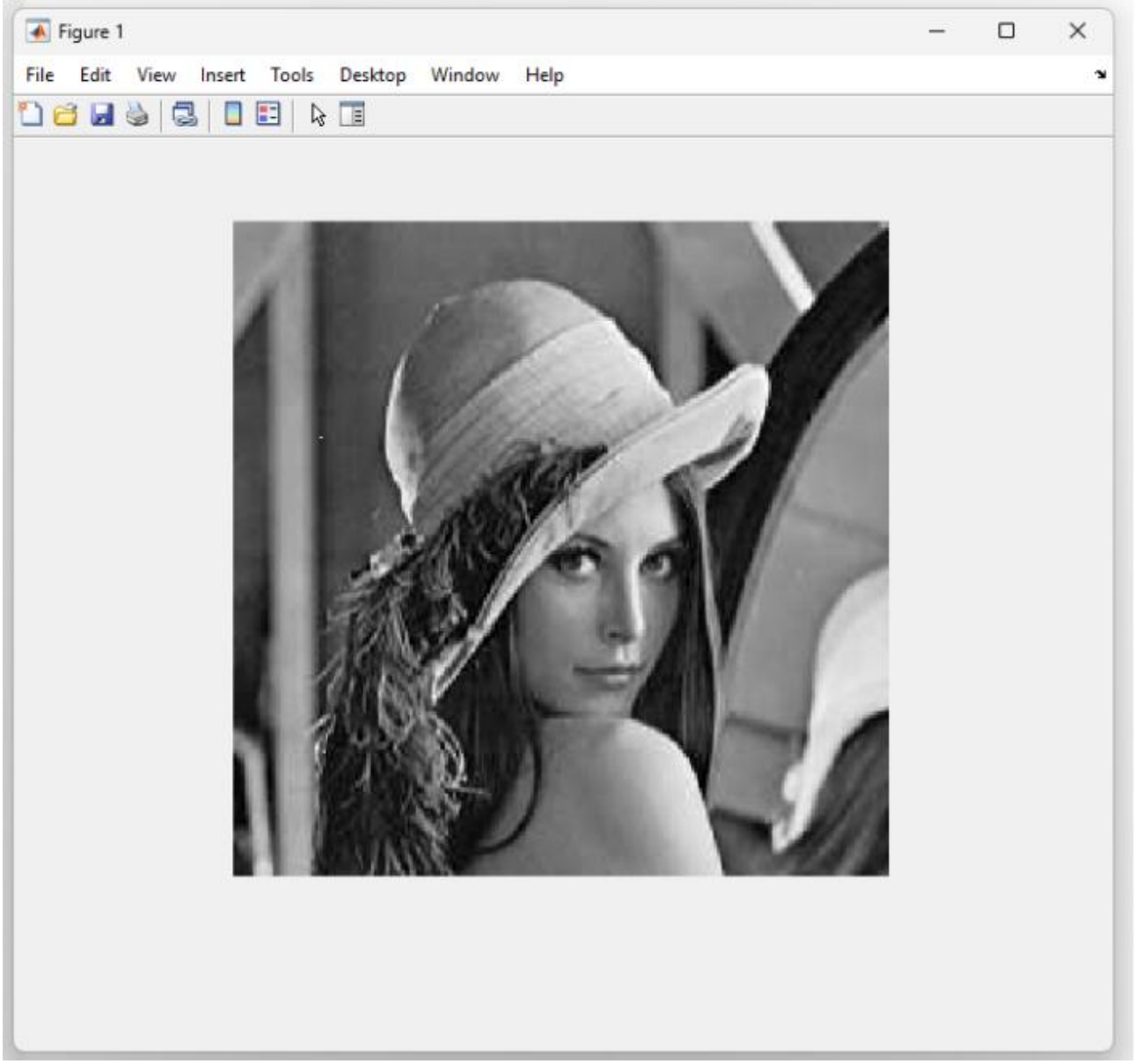
З дисципліни Комп’ютерна графіка

Виконання

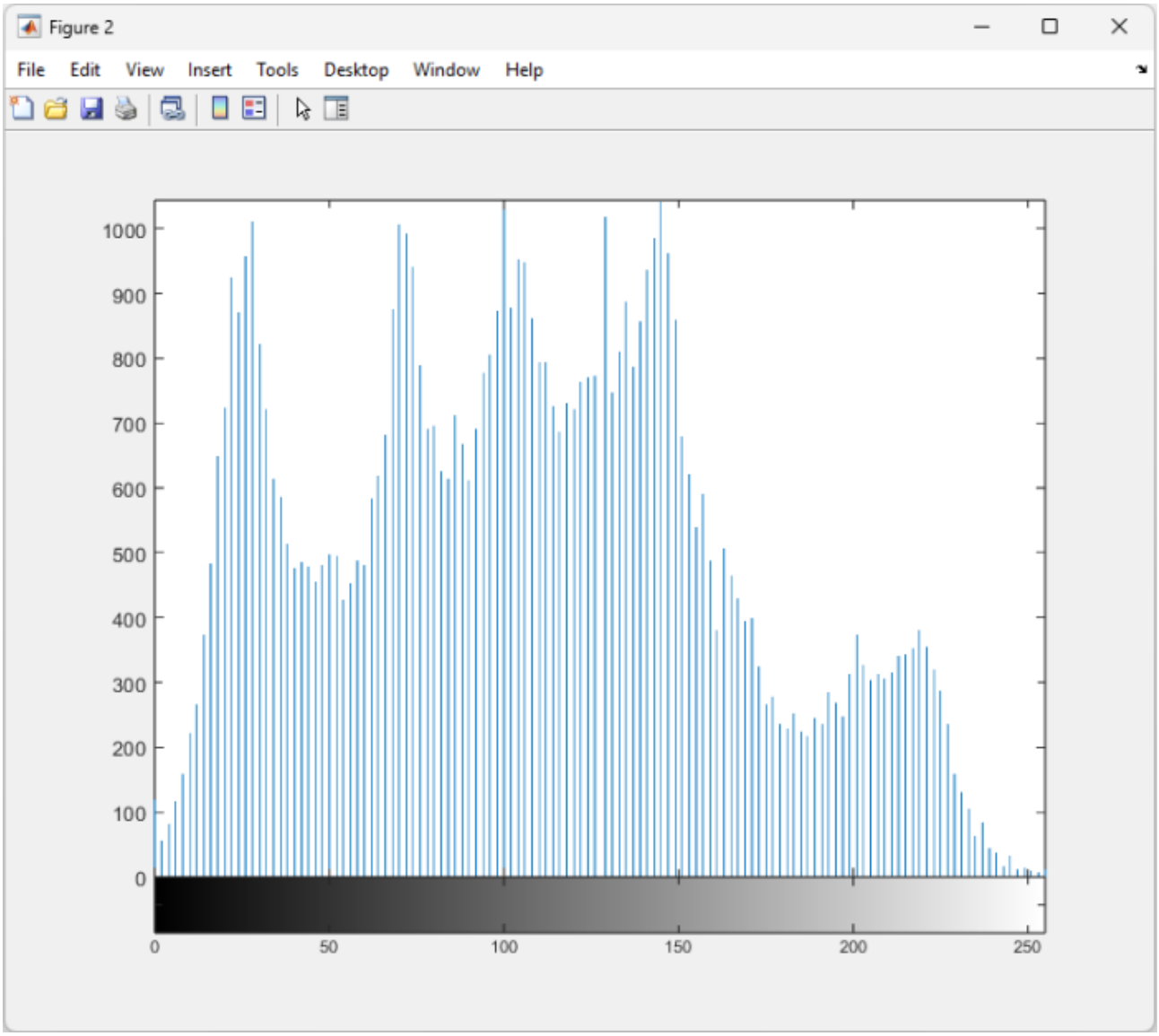
Приклад 1.1

Приклад 1.2

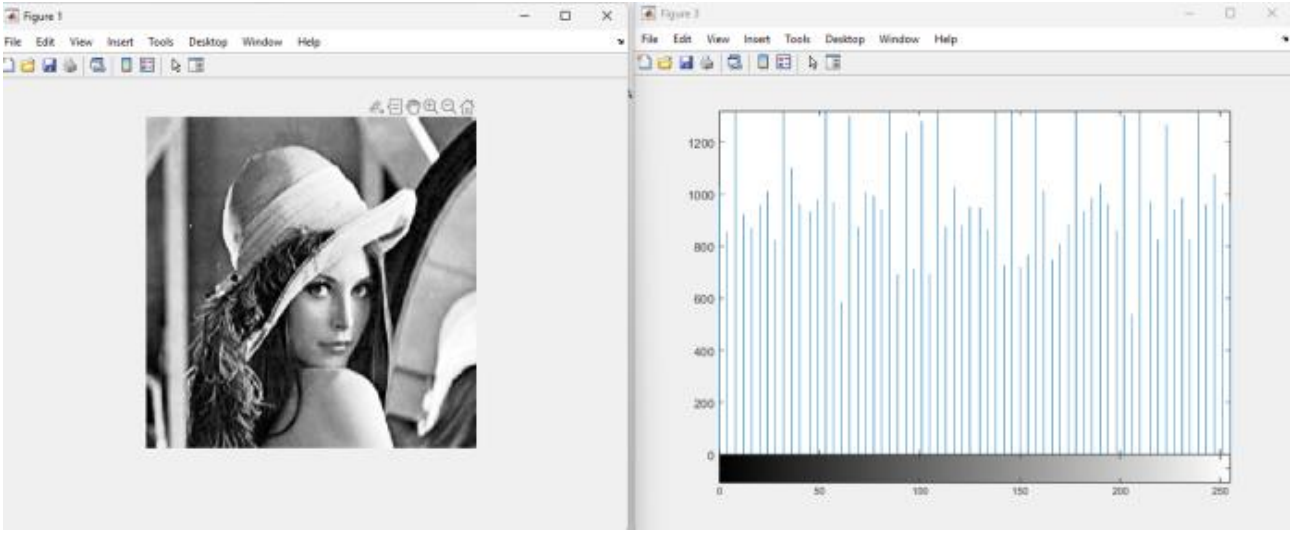
```
>> I(85,35);
>> imshow(I)
>> I(85,35)=236;
>> imshow(I)
>>
```



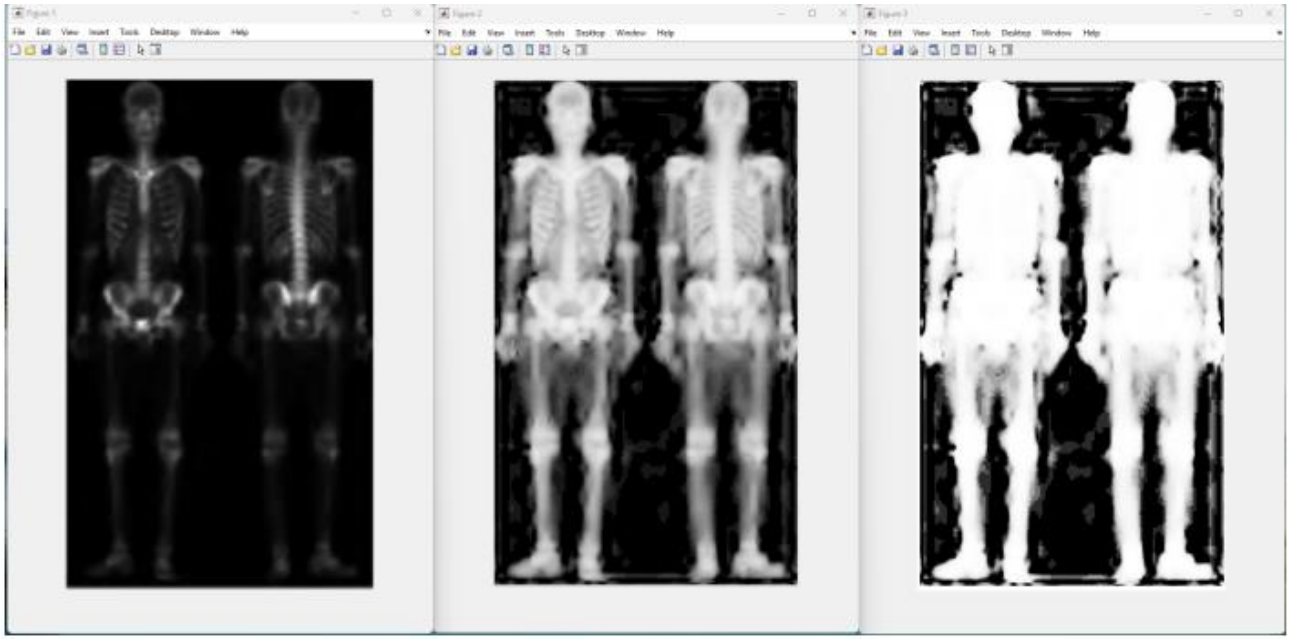
Приклад 1.3



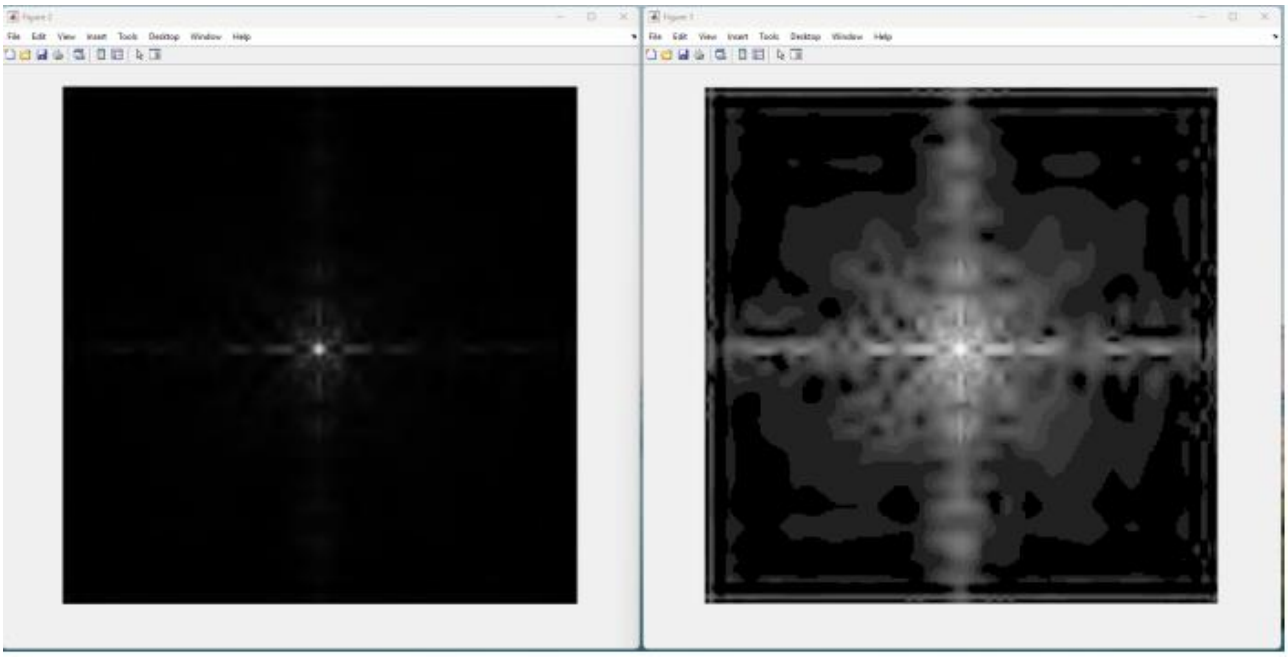
Приклад 1.4



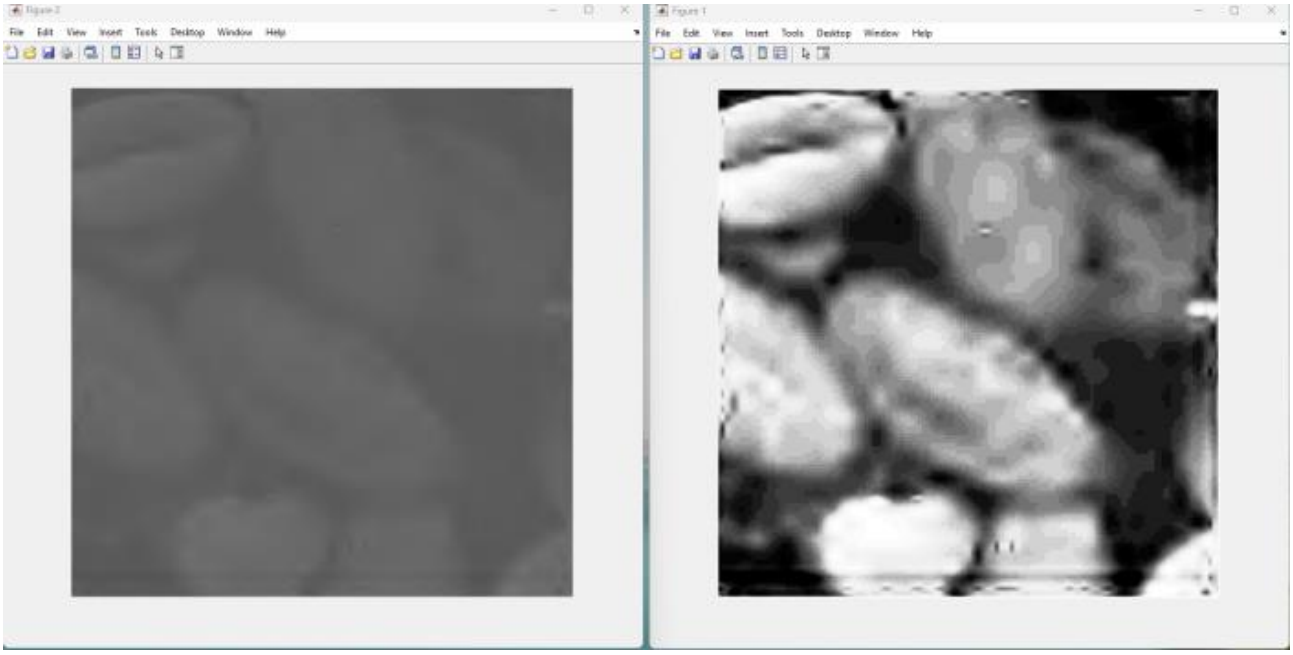
Приклад 1.5



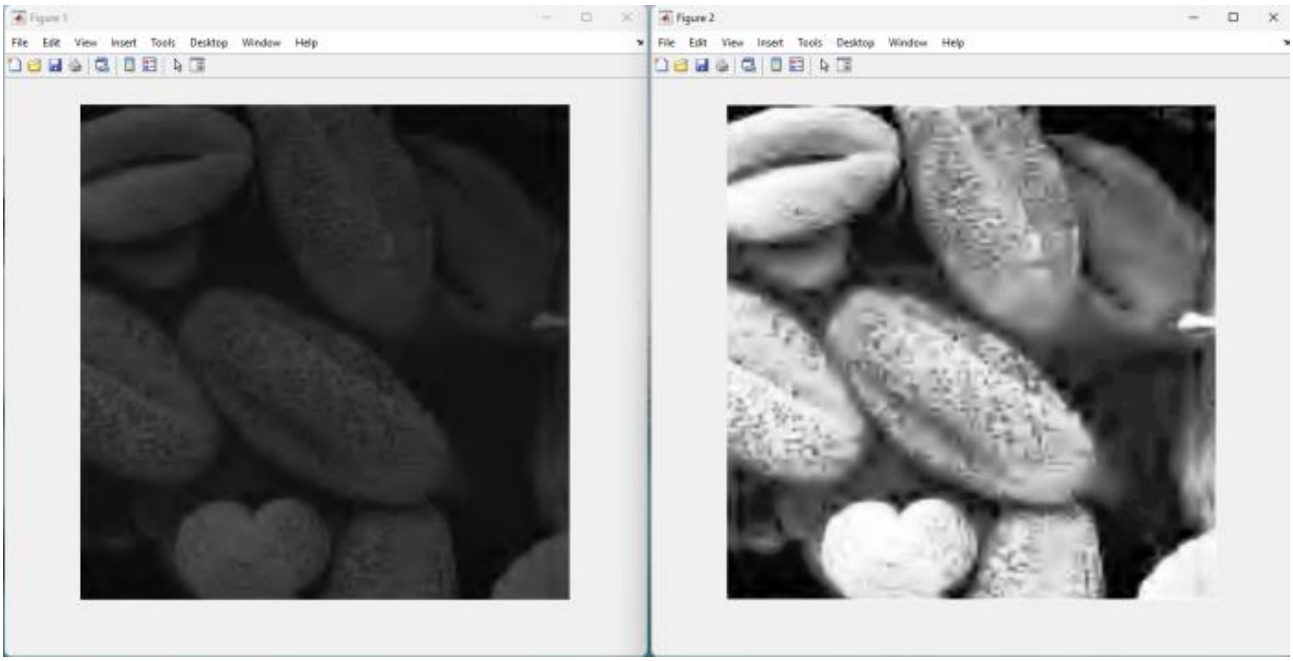
Приклад 1.6



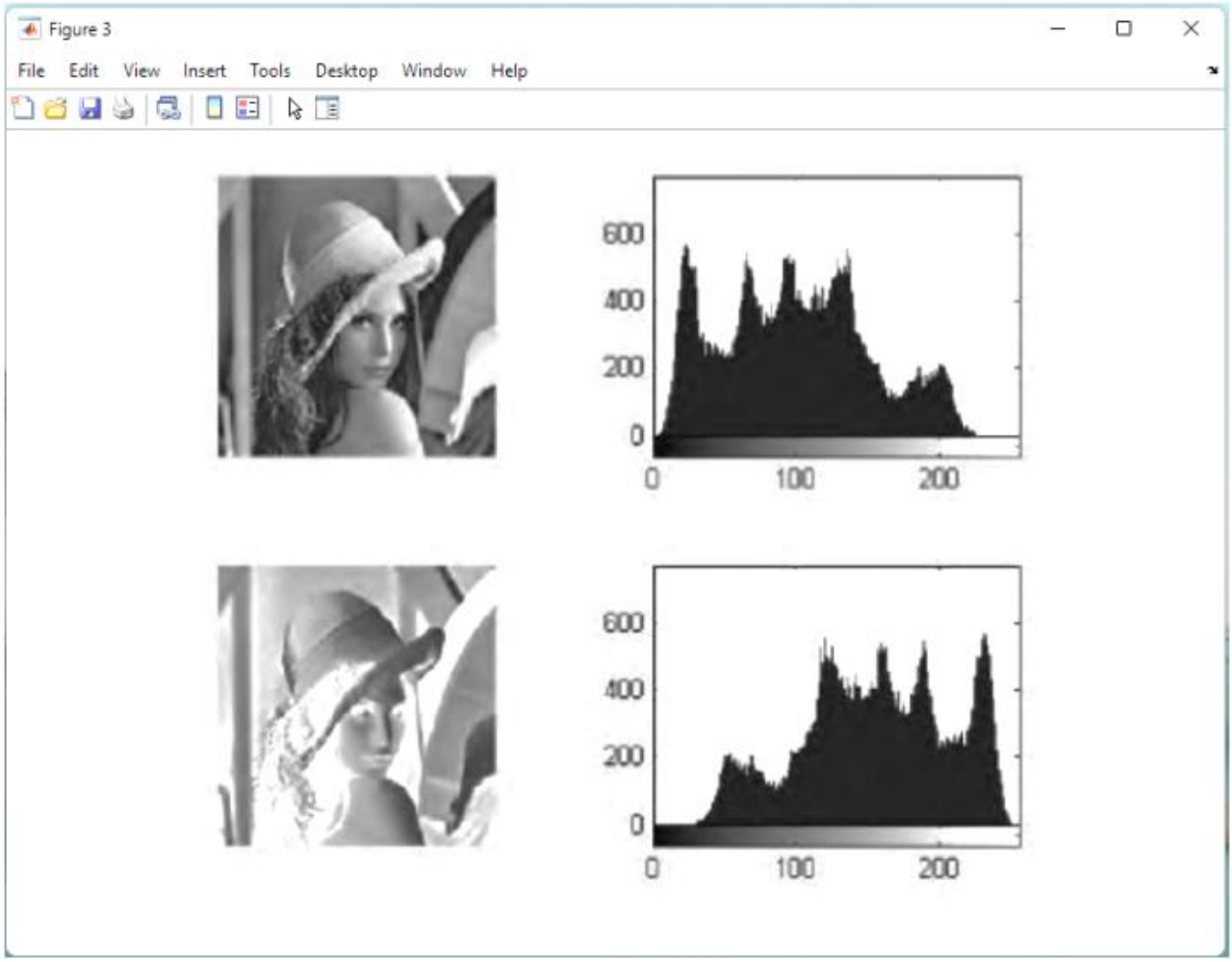
Приклад 1.7



Приклад 1.8



Приклад 1.9



Питання для самоконтролю

1. Який основний об'єкт користувацького інтерфейсу в MATLAB?
   Основним об'єктом користувацького інтерфейсу в MATLAB є "графічне вікно" або "графічний інтерфейс користувача" (GUI, Graphical User Interface). Графічне вікно є важливою частиною взаємодії користувача з MATLAB і дозволяє вам створювати і відображати різноманітні графічні об'єкти, такі як кнопки, текстові поля, графіки, таблиці тощо.

   Графічний інтерфейс користувача в MATLAB дозволяє створювати зручні програми та інтерактивні додатки для взаємодії з користувачем, що полегшує введення даних, візуалізацію результатів та виконання обчислень. MATLAB надає різні інструменти і функції для створення і налаштування графічних вікон і об'єктів GUI, такі як "GUIDE" (графічний редактор інтерфейсу користувача MATLAB) та функції, такі як "uifigure", "uibutton", "uilabel" і інші для програмування графічного інтерфейсу.
2. На які підмножини поділяються графічні об'єкти MATLAB?
   Графічні об'єкти в MATLAB поділяються на кілька основних підмножин в залежності від їхньої призначеності та функціональності. Основні підмножини графічних об'єктів в MATLAB включають:

   1. **Контейнери (Containers)**: Ця підмножина об'єктів включає в себе графічні контейнери, такі як "figure" (вікно), "uipanel" (панель), "uitabgroup" (група вкладок) та інші. Вони використовуються для розміщення та організації інших графічних об'єктів.

   2. **Кнопки та переключальні елементи (Buttons and Toggle Elements)**: Сюди входять "pushbutton" (натискна кнопка), "togglebutton" (переключальна кнопка), "radiobutton"

(радіокнопка) та інші. Вони використовуються для створення елементів управління, які реагують на кліки користувача.

3. **Текстові поля (Text Fields)**: Ця категорія включає "uilabel" (текстова мітка), "uitextarea" (текстова область) та інші об'єкти для відображення та введення текстової інформації.

4. **Графічні об'єкти (Graphics Objects)**: Вони включають графічні об'єкти, такі як "axes" (вісь), "line" (лінія), "image" (зображення) та інші для створення графіків та візуалізації даних.

5. **Списки та таблиці (Lists and Tables)**: Ця підмножина містить "uilistbox" (список), "uitable" (таблицю) та інші об'єкти для відображення списків або таблиць даних.

6. **Меню та панелі інструментів (Menus and Toolbars)**: MATLAB також має можливості для створення меню та панелей інструментів, які допомагають в організації функціональності програм та інтерфейсу.

7. **Інші спеціалізовані об'єкти**: До цієї категорії входять різноманітні спеціалізовані графічні об'єкти, такі як "uitree" (дерево), "axes3" (тривимірна вісь) та інші.

Користувачі можуть комбінувати ці різні типи графічних об'єктів для створення складних інтерфейсів та програм для взаємодії з MATLAB.

3. Яким чином можна отримати доступ до властивостей об'єкта?

Для отримання доступу до властивостей об'єкта в MATLAB використовується звертання до цих властивостей за допомогою крапки (dot notation) або функції `get`. Властивості об'єкта визначають його стан, параметри та характеристики. Ось кілька способів отримання доступу до властивостей об'єкта:

1. **Dot Notation (Крапкова нотація)**: Ви можете звертатися до властивостей об'єкта, використовуючи крапкову нотацію, де ім'я об'єкта, крапка і ім'я властивості розділені точкою. Наприклад:

```matlab
myObject.propertyName = 42;
value = myObject.propertyName;
```

У першому рядку ми встановлюємо властивість `propertyName` об'єкта `myObject` на значення 42, а в другому рядку ми отримуємо це значення.

2. **Функція `get`**: Ви можете використовувати функцію `get` для отримання значення властивостей об'єкта. Наприклад:

```matlab
value = get(myObject, 'propertyName');
```

В цьому прикладі функція `get` використовується для отримання значення властивості `propertyName` об'єкта `myObject`.

3. **Параметрів конструктору**: Іноді властивості об'єкта можуть бути встановлені під час його створення через параметри конструктору. Наприклад:

```matlab
myObject = MyClass('propertyName', 42);
```

У цьому прикладі значення властивості `propertyName` об'єкта `myObject` встановлюється під час створення об'єкта.

4. **Функція `properties`**: Ви також можете використовувати функцію `properties`, щоб отримати список доступних властивостей об'єкта. Наприклад:

```matlab
propList = properties(myObject);
```

Ця функція поверне список імен властивостей, які можуть бути використані для подальшої роботи з об'єктом.

Отримання доступу до властивостей об'єкта дозволяє вам читати та змінювати його стан, налаштувати параметри та взаємодіяти з об'єктом в MATLAB.

4. За допомогою якої команди відображається зображення з масиву на екрані?

Для відображення зображення з масиву на екрані в MATLAB використовуйте функцію `imshow`. Ось приклад використання:

```matlab
% Завантаження зображення з файлу (цей крок опціональний)
imageData = imread('image.jpg');

% Відображення зображення на екрані
imshow(imageData);
```

У цьому прикладі ми спочатку завантажуємо зображення з файлу за допомогою функції `imread` і зберігаємо його дані в змінній `imageData`. Потім ми використовуємо функцію `imshow`, щоб відобразити це зображення на екрані. Ви можете передавати `imshow` як сам масив зображення, так і шлях до файлу зображення.

5. Що потрібно ввести у командному рядку для конвертації одного зображення в інше з більш рівномірною гістограмою?

Для конвертації одного зображення в інше з більш рівномірною гістограмою в MATLAB, ви можете використовувати функцію `histeq`. Ця функція вирівнює гістограму зображення, роблячи його розподіл інтенсивності пікселів більш рівномірним.

Ось приклад використання `histeq`:

```matlab
% Завантаження зображення з файлу (цей крок опціональний)
imageData = imread('input_image.jpg');

% Конвертація зображення з більш рівномірною гістограмою
equalizedImage = histeq(imageData);

% Відображення початкового та обробленого зображень
```

```
figure;
subplot(1, 2, 1);
imshow(imageData);
title('Початкове зображення');
subplot(1, 2, 2);
imshow(equalizedImage);
title('Зображення з рівномірною гістограмою');
```

У цьому прикладі ми спочатку завантажуємо початкове зображення з файлу за допомогою функції `imread`. Потім ми використовуємо функцію `histeq`, щоб створити версію зображення з рівномірною гістограмою і зберегти її в змінній `equalizedImage`. Нарешті, ми відображаємо обидва зображення, щоб порівняти їх.

6. Якою функцією реалізується еквалізація?

Еквалізація гістограми в MATLAB реалізується функцією `histeq`. Функція `histeq` використовується для підвищення контрастності та поліпшення розподілу інтенсивності пікселів на зображенні, зробивши гістограму більш рівномірною.

Синтаксис функції `histeq` виглядає наступним чином:

```matlab
outputImage = histeq(inputImage, n);
```

- `inputImage` - це вхідне зображення, для якого ви хочете виконати еквалізацію гістограми.
- `n` (опціонально) - це кількість рівних інтервалів гістограми (за замовчуванням `n` дорівнює 64).

Функція `histeq` повертає зображення `outputImage` з відкоригованою гістограмою. Після виклику цієї функції гістограма вихідного зображення буде більш рівномірною, що поліпшить контраст та якість зображення.

7. Які дії потрібно виконати для запуску програми?

Для запуску програми в MATLAB вам слід виконати такі кроки:

1. **Запуск MATLAB**: Відкрийте MATLAB, подвійно клацнувши на його ярлику на робочому столі або запустивши його зі стартового меню.

2. **Створення або завантаження програми**: Ви можете створити новий скрипт або функцію за допомогою вбудованого текстового редактора MATLAB, або завантажити вже існуючу програму.

3. **Редагування програми**: Відкрийте програму для редагування у текстовому редакторі MATLAB. Введіть код вашої програми в цьому редакторі.

4. **Збереження програми**: Після написання коду збережіть вашу програму, використовуючи розширення файлу `.m`, наприклад, "myprogram.m".

5. **Виконання програми**: Щоб виконати програму, ви можете ввести її ім'я в командному вікні MATLAB, наприклад:

```matlab
myprogram
```

Або ж ви можете натиснути кнопку "Run" (виконати) в текстовому редакторі MATLAB, яка запустить вашу програму.

6. **Спостереження результатів**: Результати виконання програми виводяться в командному вікні MATLAB, або можуть бути відображені в графічному вікні, якщо програма створює візуалізацію або графіки.

Ці кроки допоможуть вам запустити вашу програму в MATLAB і переглянути результати її виконання.

8. В чому полягає алгоритм перетворення зображень?

Алгоритм перетворення зображень включає в себе ряд операцій та обчислень, які змінюють властивості пікселів на зображенні для досягнення певної мети, такої як поліпшення контрастності, підвищення роздільної здатності, зменшення шуму, зміна кольору та багато інших. Ось загальний алгоритм для перетворення зображень:

1. **Завантаження зображення**: Почніть з завантаження вхідного зображення, яке ви хочете обробити. Завантаження може бути виконано з файлу або створено шляхом генерації зображення або взяття його з іншого джерела.

2. **Попередня обробка (опціонально)**: Якщо необхідно, виконайте попередню обробку зображення, таку як видалення шуму, вирівнювання, роздільна здатність, попередню обрізку тощо.

3. **Виконання обраного операції**: Виконайте специфічну операцію або обчислення, які відповідають вашим цілям обробки зображення. Наприклад, це може бути еквалізація гістограми, зміна розміру, фільтрація, обробка кольору тощо.

4. **Збереження результату (опціонально)**: Збережіть оброблене зображення, якщо ви хочете зберегти результат.

5. **Відображення або аналіз результату (опціонально)**: Перегляньте або проаналізуйте результати обробки зображення, щоб впевнитися в їхній якості та відповідності ваших цілям.

Це загальний алгоритм. Зауважте, що конкретні дії та операції в алгоритмі залежать від вашої мети обробки зображення. Наприклад, для еквалізації гістограми ви використовуєте функцію `histeq`, а для фільтрації ви можете використовувати фільтри як Гаусса чи медіанний фільтр. Важливо розуміти, яку конкретну операцію потрібно виконати для досягнення вашої мети обробки зображення.

9. Як можна визначити розмір вихідного зображення?

Для визначення розміру вихідного зображення в MATLAB, ви можете використовувати функцію `imresize`, яка дозволяє змінювати розмір зображення на підставі зазначених параметрів. Синтаксис функції `imresize` виглядає так:

```matlab
outputImage = imresize(inputImage, scale);
```

- `inputImage` - це вхідне зображення, для якого ви хочете змінити розмір.

- `scale` - це масштабний фактор, що визначає, яким чином змінюється розмір зображення. Зазвичай це дійсне число:
  - Якщо `scale` більше 1, зображення збільшується (наприклад, `1.5` подвійна розмір зображення).
  - Якщо `scale` менше 1, зображення зменшується (наприклад, `0.5` половина розміру зображення).

`outputImage` буде вміщувати оброблене зображення з новим розміром.

Ось приклад зміни розміру зображення:

```matlab
% Завантаження зображення з файлу (цей крок опціональний)
inputImage = imread('input_image.jpg');

% Змінюємо розмір зображення (зменшення до половини розміру)
scale = 0.5;
outputImage = imresize(inputImage, scale);

% Відображення початкового та зміненого зображень
figure;
subplot(1, 2, 1);
imshow(inputImage);
title('Початкове зображення');
subplot(1, 2, 2);
imshow(outputImage);
title('Змінене зображення');
```

У цьому прикладі ми спочатку завантажуємо вхідне зображення і потім використовуємо функцію `imresize`, щоб зменшити його розмір до половини від оригіналу. Результат зберігається в `outputImage`, і ми відображаємо обидва зображення для порівняння.

10. Наведіть приклади використання можливостей MATLAB для покращення якості зображення.

MATLAB пропонує різні можливості для покращення якості зображень шляхом застосування різних обробок та фільтрів. Ось декілька прикладів використання можливостей MATLAB для покращення якості зображення:

1. **Еквалізація гістограми**:

   ```matlab
   % Завантаження зображення
   inputImage = imread('input_image.jpg');

   % Еквалізація гістограми для покращення контрастності
   outputImage = histeq(inputImage);

   % Відображення початкового та обробленого зображень
   figure;
   subplot(1, 2, 1);
   imshow(inputImage);
   title('Початкове зображення');
   ```

```
subplot(1, 2, 2);
imshow(outputImage);
title('Зображення з рівномірною гістограмою');
```

2. **Фільтрація зображення для зменшення шуму**:

```matlab
% Завантаження зображення
inputImage = imread('noisy_image.jpg');

% Використання медіанного фільтра для зменшення шуму
outputImage = medfilt2(inputImage);

% Відображення початкового та обробленого зображень
figure;
subplot(1, 2, 1);
imshow(inputImage);
title('Початкове зображення');
subplot(1, 2, 2);
imshow(outputImage);
title('Зображення зі зменшеним шумом');
```

3. **Збільшення роздільної здатності (розміру) зображення**:

```matlab
% Завантаження зображення
inputImage = imread('low_resolution_image.jpg');

% Збільшення розміру зображення вдвічі з інтерполяцією
outputImage = imresize(inputImage, 2);

% Відображення початкового та зміненого зображень
figure;
subplot(1, 2, 1);
imshow(inputImage);
title('Початкове зображення');
subplot(1, 2, 2);
imshow(outputImage);
title('Збільшене зображення');
```

4. **Кольорова корекція**:

```matlab
% Завантаження зображення
inputImage = imread('color_image.jpg');

% Виконання корекції кольору (наприклад, зміна балансу білого)
outputImage = imadjust(inputImage, [0.2 0.6], [], 1.5);
```

```
% Відображення початкового та обробленого зображень
figure;
subplot(1, 2, 1);
imshow(inputImage);
title('Початкове зображення');
subplot(1, 2, 2);
imshow(outputImage);
title('Зображення з кольоровою корекцією');
```

Це лише декілька прикладів можливостей покращення якості зображень в MATLAB. Існує багато інших методів і технік для редагування та покращення зображень, які ви можете застосовувати в залежності від вашої конкретної задачі та цілей.