МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИСТЕТ
«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»
Кафедра програмної інженерії та інформаційних технологій управління

ЗВІТ ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ №3
З ДИСЦИПЛІНИ «ОБРОБКА ДАНИХ PYTHON»

ВИКОНАВ:
студент групи КН-220с
Кравченко Максим
ПЕРЕВІРИВ:
Коваленко С.М

Харків – 2021

Тема: Основи роботи з бібліотекою Matplotlib.

Мета: отримати базові знання та навички по роботі з бібліотекою Matplotlib.

Результати виконання лабораторної роботи

Завдання:

1. У відповідності до номеру в журналі академічної групи обрати номер індивідуального завдання.

2. Набрати рівняння за допомогою мови LaTex в клітинці блокнота.

3. Побудувати кожну лінію на окремому графіку, розмістивши їх поруч.

- всі прямі повинні бути різного кольору, але не використовувати системну послідовність кольорів (синій, помаранчевий, зелений...);
- всі прямі повинні мати різний тип ліній (пунктирна, точка тире тощо);
- всі графіки повинні мати сумісну вісь ординат;
- кожен графік повинен мати підпис;

4. Розмістити всі лінії на одному рисунку.

- всі прямі повинні бути різного кольору, але не використовувати системну послідовність кольорів (синій, помаранчевий, зелений...);
- всі прямі повинні мати різний тип ліній (пунктирна, точка тире тощо);
- підібрати масштаб таким чином, щоб всі три точки перетину прямих були в області видимості;
- додати легенду на графік;
- додати сітку, в якій задати колір та тип ліній;
- змінити розмір рисунку (наприклад, 8x16 дюймів) та розподільчу здатність (наприклад, 100 dpi);
- зробити підписи рівнянь прямих вздовж лінії з відповідним нахилом;

- додати підписи осей та назву графіку;

- заповнити кольором область, що утворена перетином всіх прямих.

5. За допомогою підмодуля numpy.linalg знайти точки перетину всіх пар прямих, відмітити їх та зробити відповідні вказівки на графіку.

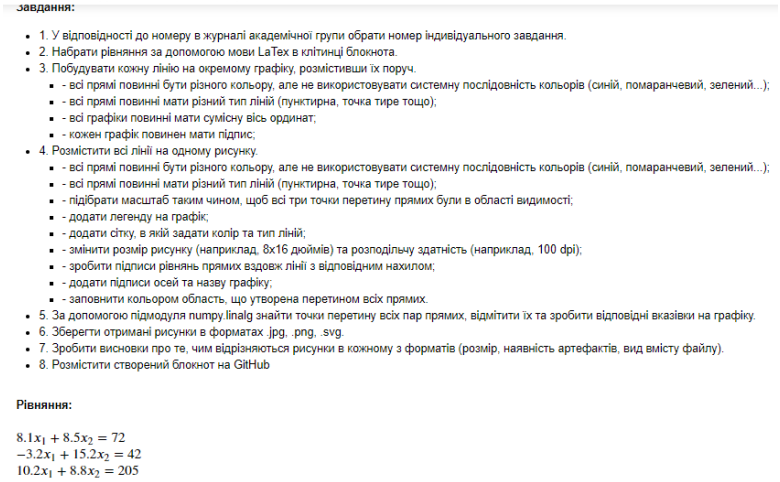
6. Зберегти отримані рисунки в форматах .jpg, .png, .svg.

7. Зробити висновки про те, чим відрізняються рисунки в кожному з форматів (розмір, наявність артефактів, вид вмісту файлу).

8. Розмістити створений блокнот на GitHub

Вар. 5

$$\begin{cases} 8.1x_1 + 8.5x_2 = 72 \\ -3.2x_1 + 15.2x_2 = 42 \\ 10.2x_1 + 8.8x_2 = 205 \end{cases}$$



Завдання:

- 1. У відповідності до номеру в журналі академічної групи обрати номер індивідуального завдання.
- 2. Набрати рівняння за допомогою мови LaTex в клітинці блокнота.
- 3. Побудувати кожну лінію на окремому графіку, розмістивши їх поруч.
  - - всі прямі повинні бути різного кольору, але не використовувати системну послідовність кольорів (синій, помаранчевий, зелений...);
  - - всі прямі повинні мати різний тип ліній (пунктирна, точка тире тощо);
  - - всі графіки повинні мати сумісну вісь ординат;
  - - кожен графік повинен мати підпис;
- 4. Розмістити всі лінії на одному рисунку.
  - - всі прямі повинні бути різного кольору, але не використовувати системну послідовність кольорів (синій, помаранчевий, зелений...);
  - - всі прямі повинні мати різний тип ліній (пунктирна, точка тире тощо);
  - - підібрати масштаб таким чином, щоб всі три точки перетину прямих були в області видимості;
  - - додати легенду на графік;
  - - додати сітку, в якій задати колір та тип ліній;
  - - змінити розмір рисунку (наприклад, 8x16 дюймів) та розподільчу здатність (наприклад, 100 dpi);
  - - зробити підписи рівнянь прямих вздовж лінії з відповідним нахилом;
  - - додати підписи осей та назву графіку;
  - - заповнити кольором область, що утворена перетином всіх прямих.
- 5. За допомогою підмодуля numpy.linalg знайти точки перетину всіх пар прямих, відмітити їх та зробити відповідні вказівки на графіку.
- 6. Зберегти отримані рисунки в форматах .jpg, .png, .svg.
- 7. Зробити висновки про те, чим відрізняються рисунки в кожному з форматів (розмір, наявність артефактів, вид вмісту файлу).
- 8. Розмістити створений блокнот на GitHub

Рівняння:

$8.1x_1 + 8.5x_2 = 72$
$-3.2x_1 + 15.2x_2 = 42$
$10.2x_1 + 8.8x_2 = 205$

Рисунок 1 – Завдання

Завдання 1

1) Лістинг коміроки:

```
import matplotlib.pyplot as plt
import numpy as np
fig, ax = plt.subplots(1, 3, figsize=(10, 4))
x = np.linspace(0, 10, 100)
fig.suptitle("Все прямые")
ax[0].set_title(str("$8.1*x_1+8.5*x_2=72$"))
ax[0].plot(x, (72-8.1*x)/8.5, color='#FF0000', linestyle='--')
ax[1].set_title(str("$−3.2*x_1+15.2*x_2=42$"))
ax[1].plot(x, (42+3.2*x)/15.2, color='#006400', linestyle='-.')
ax[2].set_title(str("$10.2*x_1+8.8*x_2=205$"))
ax[2].plot(x, (205-10.2*x)/8.8, color='#00008B', linestyle=':')
```

```
ax[0].set_ylim([0, 20]);
ax[2].set_ylim([0, 20]);
ax[1].get_yaxis().set_ticklabels([]);
ax[2].get_yaxis().set_ticklabels([]);
```

2) Результат завдання:

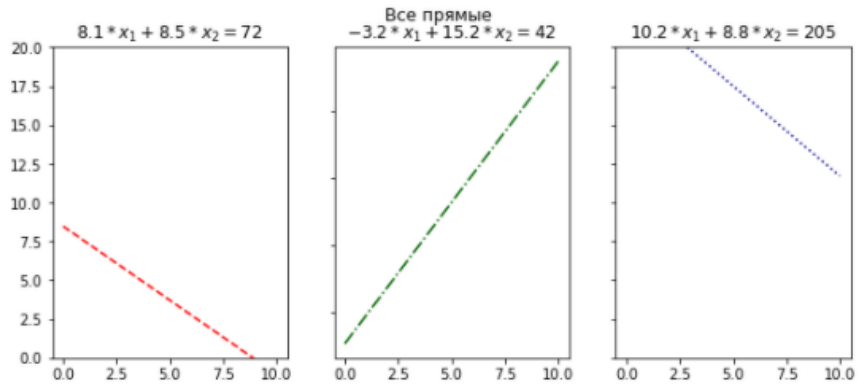


Рисунок 2 – Результат завдання 1

Завдання 2

1) Лістинг комірки:

```
from numpy import linalg as LA
x = np.linspace(0, 80, 80)
plt.figure(figsize=(16, 8),dpi=100) # я специально сделал 16*8 так как рисунок не красивый если наоборот и ничего не понятно на нем
plt.plot(x, (72-8.1*x)/8.5, color='#FF0000', linestyle='--',label='$f_1$')
plt.plot(x, (42+3.2*x)/15.2, color='#006400', linestyle='-.',label='$f_2$')
plt.plot(x, (205-10.2*x)/8.8, color='#00008B', linestyle=':', label='$f_3$')
x2=x[15:73]
x3=x[5:16]
plt.fill_between(x2, (72-8.1*x2)/8.5,(205-10.2*x2)/8.8 , color='#C0C0C0')
plt.fill_between(x3, (72-8.1*x3)/8.5,(42+3.2*x3)/15.2 , color='#C0C0C0')
plt.xlabel('$x_2$', fontsize=12)
plt.ylabel('$x_1$', fontsize=12)
plt.xlim(0, 80)
plt.grid(color='#C0C0C0', linestyle='-')
plt.title("Все прямые на одном рисунке")
plt.text(20, 10, '$−3.2*x_1+15.2*x_2=42$',rotation=4)
plt.text(25, -15, '$10.2*x_1+8.8*x_2=205$',rotation=-25)
plt.text(15, -13, '$8.1*x_1+8.5*x_2=72$',rotation=-20)
plt.legend();
a1 = [[8.1, 8.5],[-3.2, 15.2]]
b1 = [72, 42]
x4 = LA.solve(a1, b1)
plt.plot(x4[0], x4[1], marker="o", c="g")
a2 = [[8.1, 8.5],[10.2, 8.8]]
b2 = [72, 205]
x4 = LA.solve(a2, b2)
plt.plot(x4[0], x4[1], marker="o", c="r")
a3 = [[10.2, 8.8],[-3.2, 15.2]]
b3 = [205, 42]
```

```
x4 = LA.solve(a3, b3)
plt.plot(x4[0], x4[1], marker="o", c="b")
plt.arrow(5, -15, 0, 15, length_includes_head=True, head_width=1, head_length=2)
plt.arrow(72, -40, 0, -15, length_includes_head=True, head_width=1, head_length=2)
plt.arrow(15, 17, 0, -10, length_includes_head=True, head_width=1, head_length=2)
plt.text(8, 20, '$Точка\quadпересичения\quadf_2\quadand\quadf_3$',rotation=0)
plt.text(2, -20, '$Точка\quadпересичения\quadf_1\quadand\quadf_2$',rotation=0)
plt.text(60, -38, '$Точка\quadпересичения\quadf_1\quadand\quadf_3$',rotation=0)
plt.savefig('my_figure.png', dpi=200)
plt.savefig('my_figure.jpg', dpi=200)
plt.savefig('my_figure.svg', dpi=200)
plt.show();
```

2) Результат завдання:

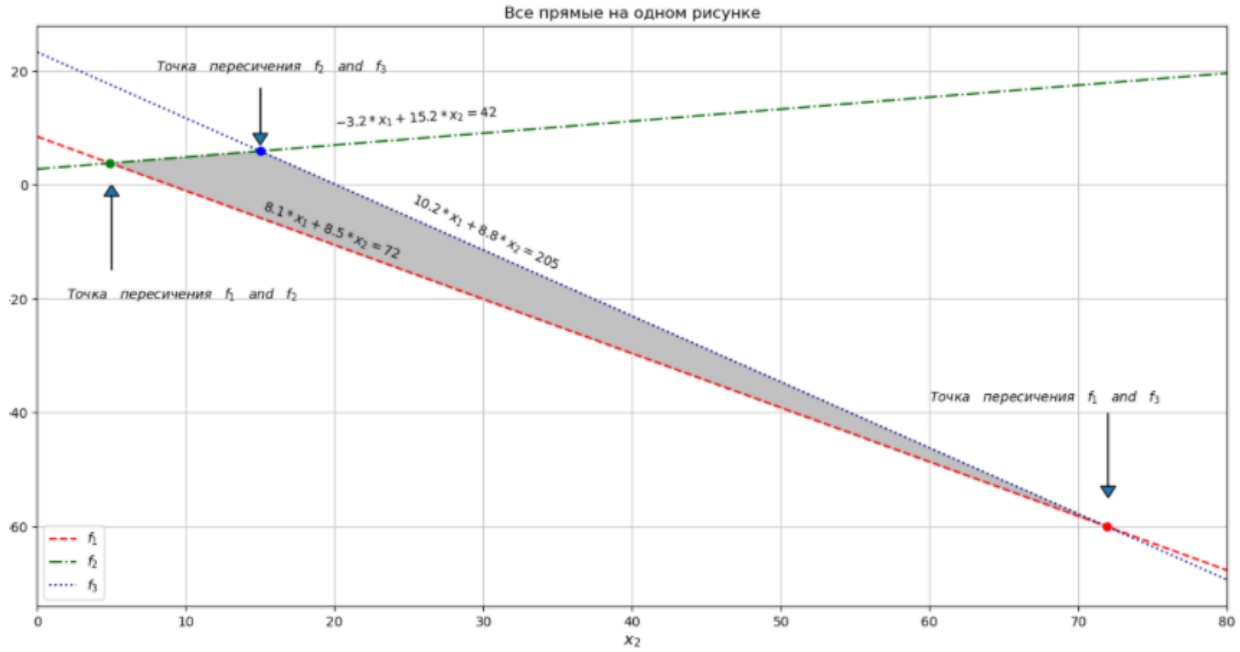


Рисунок 3 – Результат завдання 2

Висновок, щодо відмінностей форматів збережених картинок:

JPEG (він же JPG) - це формат зображень, який використовує стиснення з втратами і не підтримує прозорість. Дозволяє налаштовувати рівень якості зберігається зображення - при його зниженні видаляються деталі і додаються шуми на зображення, проте розмір стає більш компактним.

PNG 24 - це формат зображень, який працює з кольоровими зображеннями, використовує стиснення без втрат і дозволяє зберігати прозорість. Налаштувати якість збереження в PNG 24 неможливо, однак,

можна адаптувати зберігається зображення для досягнення мінімального розміру файлу: для цього можна знизити кількість квітів в зображенні.

На відміну від двох форматів вище SVG (Scalable Vector Graphics) - не чисто растровий формат. Це векторний формат, близький до AI в Adobe Illustrator і EPS. Векторна графіка поступово набуває популярності в мережі і у UI дизайнерів.

Іноді зручно представляти SVG як «HTML для ілюстрацій». Цей формат трохи відрізняється від інших.

SVG найкраще підходить для відображення логотипів, іконок, карт, прапорів, графіків і іншої графіки, створеної в векторних графічних редакторах типу Illustrator, Sketch і Inkscape. SVG написаний на XML розмітки, його можна редагувати в будь-якому текстовому редакторі, а також за допомогою JS і CSS. Векторна графіка масштабується під будь-який розмір без втрати якості, що ідеально підходить для адаптивного дизайну.

Отже, фотографії та зображення з великою кількістю квітів найкраще зберігати в JPEG. Але варто пам'ятати, що алгоритми компресії JPEG стискає зображення з втратою якості. Іконки, схеми, картинки з великою кількістю тексту і зображення з прозорістю оптимальніше зберігати в PNG 24. алгоритми компресії PNG 24 стискає зображення без втрати якості або використовувати SVG.

Посилання на створений блокнот, розміщений на Github та nbviewer.

1) https://github.com/maksim19-06-1/lab_python/blob/main/lab3/lab3.ipynb
2) https://nbviewer.jupyter.org/github/maksim19-06-1/lab_python/blob/main/lab3/lab3.ipynb

Висновки

В результаті виконання даної лабораторної роботи ознайомилися з особливістю використання мови Python в середовищі Jupyter Notebook та отримали навички роботи з бібліотекою Matplotlib. Та виконаний проєкт розмістили на Github. Та зробили висновок, щодо відмінностей форматів збережених картинок.