Міністерство освіти та науки України
Львівський національний університет імені Івана Франка

Факультет електроніки та
Комп’ютерних технологій

Звіт

Про виконання лабораторної роботи №3
“Операції над нечіткими множинами”

Виконав:
Студент групи ФеІ-44
Сапанюк М.І.
Перевірила:
Притула М.

Львів 2022

Мета:

Ознайомитися з основними операціями над нечіткими множинами та їх властивостями.

Хід роботи:

1. Задати функції приналежності нечітких множин A = {низька людина} і B = {висока людина}, заданих на універсумі [150, 200]. Побудувати їх графіки.

2. Запрограмувати реалізацію операцій доповнення, перетину, об’єднання, різниці, симетричної різниці, концентрування та розтягування нечітких множин.

3. За допомогою операцій над нечіткими множинами побудувати графіки функцій приналежності, які характеризують висловлювання “низька і висока людина”, “низька або висока людина”, “невисока людина”, “злегка низька людина”, “дуже висока людина”.

4. Перевірити виконання основних властивостей операцій над нечіткими множинами: комутативності, асоціативності, дистрибутивності, інволюції та виконання законів де Моргана.

Виконання завдання:

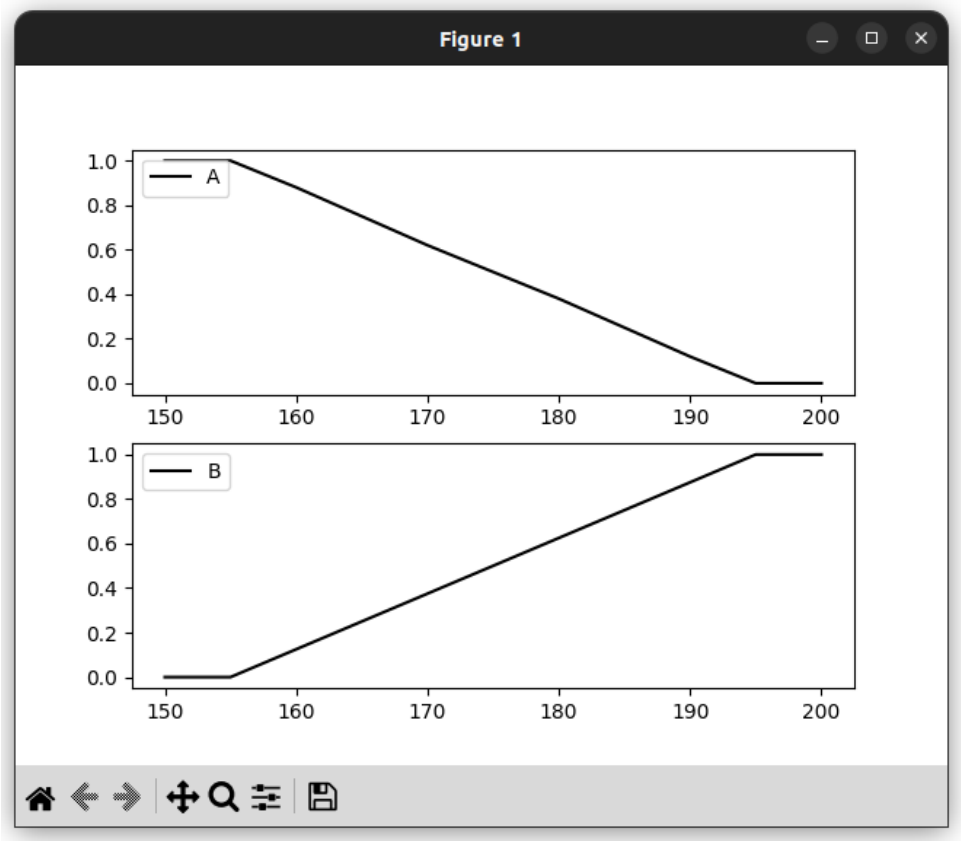
Figure 1
A
B
1.0
0.8
0.6
0.4
0.2
0.0
150
160
170
180
190
200

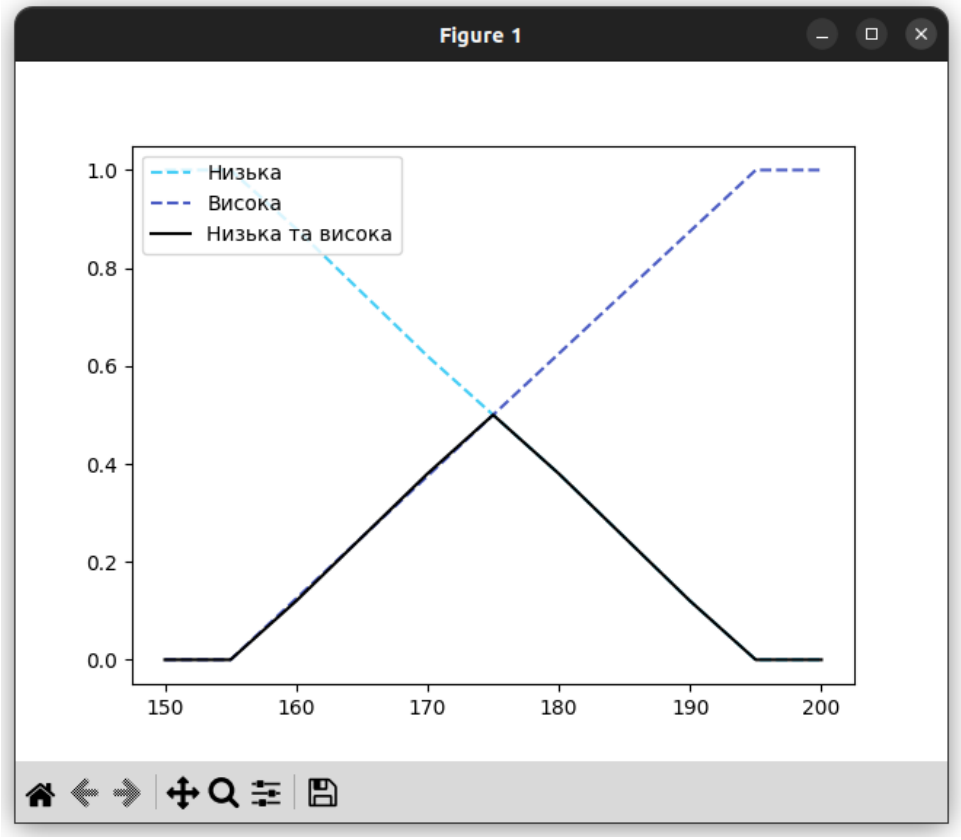
Figure 1
Низька
Висока
Низька та висока
1.0
0.8
0.6
0.4
0.2
0.0
150
160
170
180
190
200

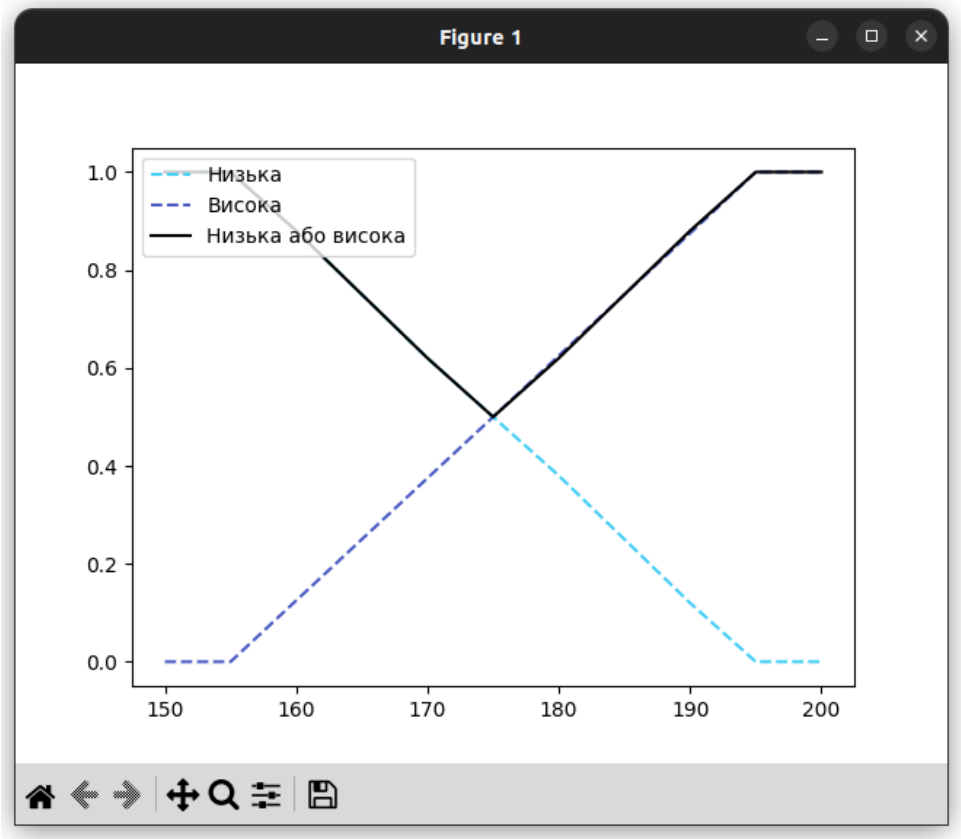
Figure 1
Низька
Висока
Низька або висока
1.0
0.8
0.6
0.4
0.2
0.0
150
160
170
180
190
200

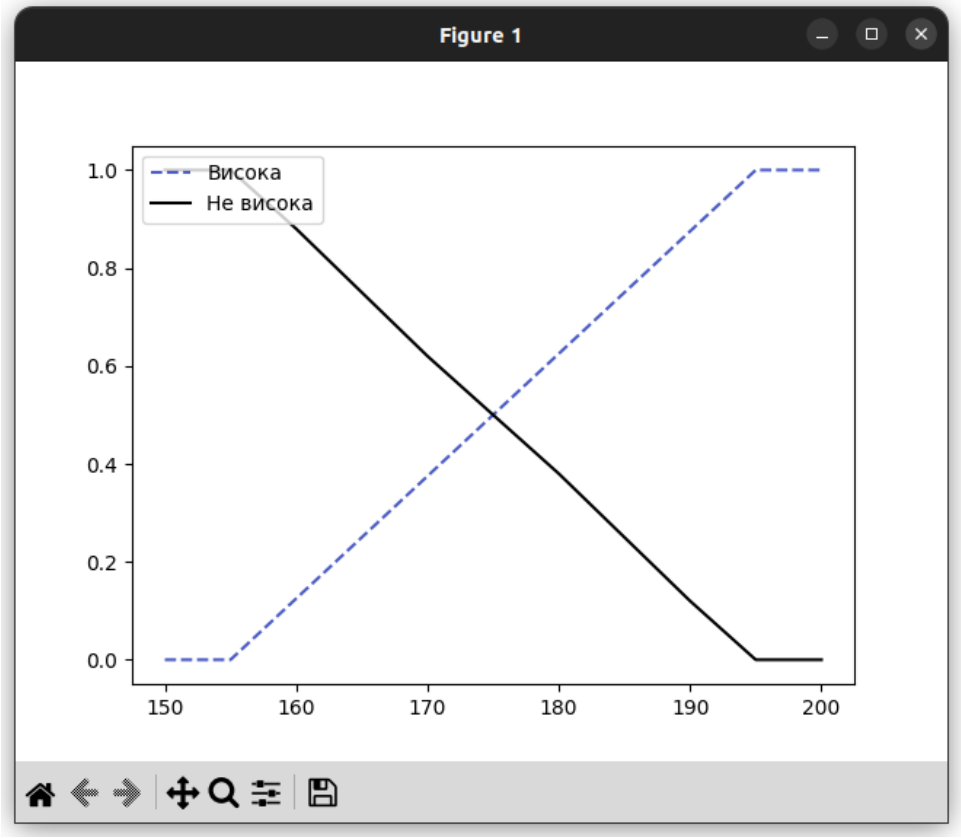
Figure 1
Висока
Не висока
1.0
0.8
0.6
0.4
0.2
0.0
150
160
170
180
190
200

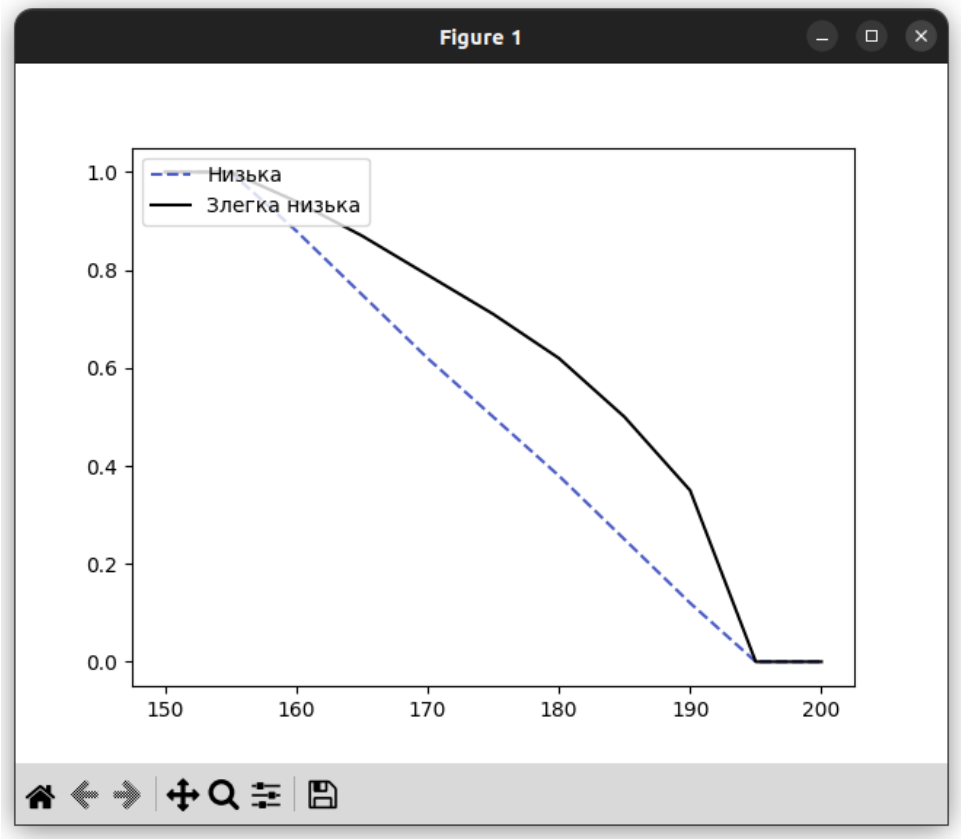
Figure 1
Низька
Злегка низька
1.0
0.8
0.6
0.4
0.2
0.0
150
160
170
180
190
200

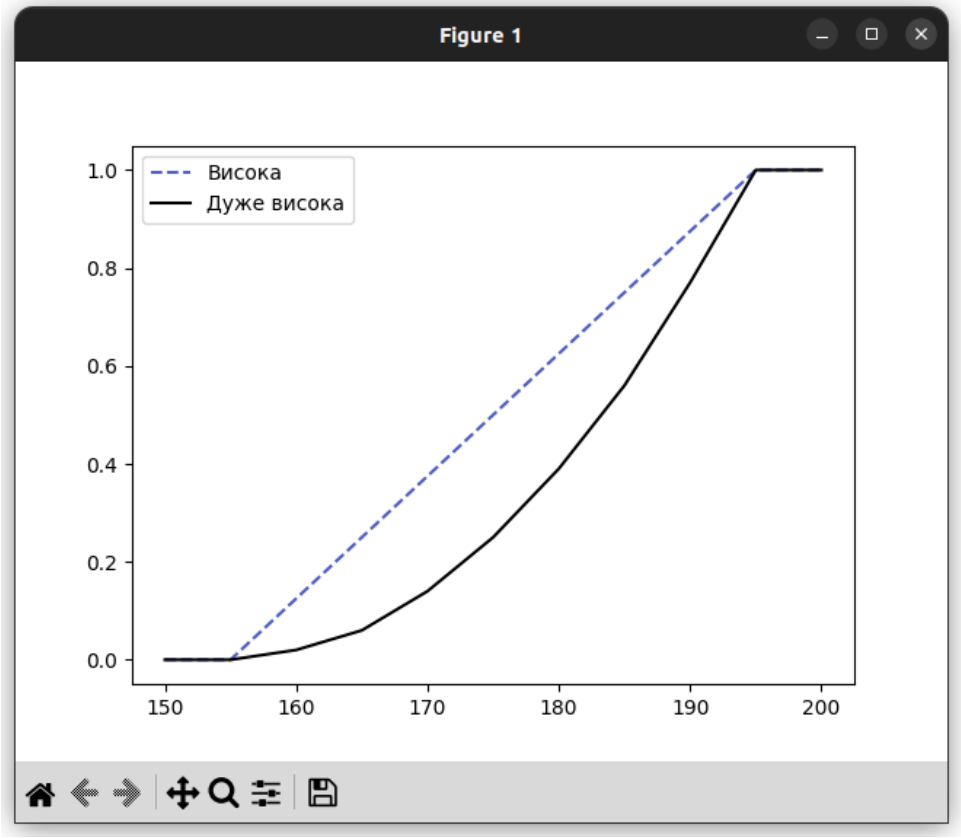
Figure 1
Висока
Дуже висока
1.0
0.8
0.6
0.4
0.2
0.0
150
160
170
180
190
200

```
Вектори
A - низька [1, 1, 0.88, 0.75, 0.62, 0.5, 0.38, 0.25, 0.12, 0, 0]
B - Висока [0, 0, 0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875, 1, 1]

Part 2
Доповнення [0, 0, 0.12, 0.25, 0.38, 0.5, 0.62, 0.75, 0.88, 1, 1]
Доповнення [1, 1, 0.88, 0.75, 0.62, 0.5, 0.38, 0.25, 0.12, 0, 0]
Об'єднання [1, 1, 0.88, 0.75, 0.62, 0.5, 0.62, 0.75, 0.88, 1, 1]
Перетин [0, 0, 0.12, 0.25, 0.38, 0.5, 0.38, 0.25, 0.12, 0, 0]
Різниця [1, 1, 0.88, 0.75, 0.62, 0.5, 0.38, 0.25, 0.12, 0, 0]
Симетрична Різниця [1, 1, 0.88, 0.75, 0.62, 0.5, 0.62, 0.75, 0.88, 1, 1]
Концентрація [1, 1, 0.77, 0.56, 0.38, 0.25, 0.14, 0.06, 0.01, 0, 0]
Розтягування [1.0, 1.0, 0.94, 0.87, 0.79, 0.71, 0.62, 0.5, 0.35, 0.0, 0.0]

Частина 4
A [1, 1, 0.88, 0.75, 0.62, 0.5, 0.38, 0.25, 0.12, 0, 0]
B [0, 0, 0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875, 1, 1]
Комутативність True
Деморгана True
C [0.0, 0.3, 0.5, 0.7, 0.9, 1, 0.8, 0.6, 0.4, 0.2, 0.0]
Асоциативність True
Дистрибутивності True
```

Висновок:

Виконавши лабораторну роботу, ознайомився з основними операціями над нечіткими множинами та їх властивостями, та запрограмував їх.

Додаток:

```python
from matplotlib import pyplot as plt


def addition(array: list[float]) -> list[float]:
    temp: list[float] = [round(1 - x, 2) for x in array]
    return temp


def union(array1: list[float], array2: list[float]) -> list[float]:
    temp: list[float] = [round(array1[i], 2) if array1[i] > array2[i] else round(array2[i], 2) for i in
                         range(0, len(array1))]
    return temp


def intersection(array1: list[float], array2: list[float]) -> list[float]:
    temp: list[float] = [round(array1[i], 2) if array1[i] < array2[i] else round(array2[i], 2) for i in
                         range(0, len(array1))]
    return temp


def diff(array1: list[float], array2: list[float]) -> list[float]:
    return intersection(array1, addition(array2))


def scale2(array1: list[float]) -> list[float]:
    temp: list[float] = [round(x ** 2, 2) for x in array1]
    return temp
```

```python
def scale05(array1: list[float]) -> list[float]:
    temp: list[float] = [round(x ** 0.5, 2) for x in array1]
    return temp


def aprocZ(x: int, a: int, b: int) -> float:
    if x <= a:
        return 1
    elif a < x < b:
        return round((b - x) / (b - a), 2)
    else:
        return 0


def aprocS(x: int, a: int, b: int) -> float:
    if x <= a:
        return 0
    elif a < x < b:
        return (x - a) / (b - a)
    else:
        return 1


def part2(array_mu1: list[float], array_mu2: list[float]) -> None:
    print("\nPart 2")
    print("Доповнення", addition(array_mu1))
    print("Доповнення", addition(array_mu2))
    print("Об'єднання", union(array_mu1, array_mu2))
    print("Перетин", intersection(array_mu1, array_mu2))
    print("Різниця", diff(array_mu1, array_mu2))
    print("Симетрична Різниця", diff(union(array_mu1, array_mu2), intersection(array_mu1, array_mu2)))
    print("Концентрація", scale2(array_mu1))
    print("Розтягування", scale05(array_mu1))


def part3(array_label: list[int], array_mu1: list[float], array_mu2:
list[float]) -> None:
    plt.plot(array_label, array_mu1, color='#44cef6', linestyle="--", label="Низька")
    plt.plot(array_label, array_mu2, color='#4b5cc4', linestyle="--", label="Висока")
    plt.plot(array_label, intersection(array_mu2, array_mu1), color="#000000", label="Низька та висока")
    plt.legend(loc=2)
    plt.show()
    plt.plot(array_label, array_mu1, color='#44cef6', linestyle="--", label="Низька")
    plt.plot(array_label, array_mu2, color='#4b5cc4', linestyle="--", label="Висока")
    plt.plot(array_label, union(array_mu1, array_mu2), color="#000000", label="Низька або висока")
    plt.legend(loc=2)
    plt.show()
    plt.plot(array_label, array_mu2, color='#4b5cc4', linestyle="--", label="Висока")
    plt.plot(array_label, addition(array_mu2), color="#000000", label="Не висока")
    plt.legend(loc=2)
    plt.show()
    plt.plot(array_label, array_mu1, color='#4b5cc4', linestyle="--", label="Низька")
    plt.plot(array_label, scale05(array_mu1), color="#000000", label="Злегка низька")
    plt.legend(loc=2)
    plt.show()
    plt.plot(array_label, array_mu2, color='#4b5cc4', linestyle="--", label="Висока")
    plt.plot(array_label, scale2(array_mu2), color="#000000", label="Дуже висока")
    plt.legend(loc=2)
    plt.show()


def commutativity(array_mu1: list[float], array_mu2: list[float]) -> bool:
    temp: list[float] = intersection(array_mu1, array_mu2)
```

```
    temp1: list[float] = intersection(array_mu2, array_mu1)
    temp2: list[float] = union(array_mu2, array_mu1)
    temp3: list[float] = union(array_mu1, array_mu2)
    return temp == temp1 and temp2 == temp3


def demorgan(array_mu1: list[float], array_mu2: list[float]) -> bool:
    array_mu1_adit: list[float] = addition(array_mu1)
    array_mu2_adit: list[float] = addition(array_mu2)
    return addition(intersection(array_mu1, array_mu2)) == union(array_mu1_adit, array_mu2_adit) and addition(
        union(array_mu1, array_mu2)) == intersection(array_mu1_adit, array_mu2_adit)


def associativity(array_mu1: list[float], array_mu2: list[float], array_mu3:
list[float]) -> bool:
    return intersection(intersection(array_mu1, array_mu2), array_mu3) == intersection(array_mu1,
                                                                                       intersection(array_mu2,
                                                                                                    array_mu3)) and union(
        union(array_mu1, array_mu2), array_mu3) == union(array_mu1, union(array_mu2, array_mu3))


def distributivity(array_mu1: list[float], array_mu2: list[float], array_mu3:
list[float]) -> bool:
    return intersection(array_mu1, union(array_mu2, array_mu3)) == union(intersection(array_mu1, array_mu2),
                                                                         intersection(array_mu1, array_mu3)) and union(
        array_mu1, intersection(array_mu2, array_mu3)) == intersection(union(array_mu1, array_mu2),
                                                                       union(array_mu1, array_mu3))


def part4(array_mu1: list[float], array_mu2: list[float]) -> None:
    print("\nЧастина 4")
    print("A", array_mu1)
    print("B", array_mu2)
    print("Комутативність", commutativity(array_mu1, array_mu2))
    print("Деморгана", demorgan(array_mu1, array_mu2))
    array_mu3: list[float] = [0.0, 0.3, 0.5, 0.7, 0.9, 1, 0.8, 0.6, 0.4, 0.2, 0.0]
    print("C", array_mu3)
    print("Асоциативність", associativity(array_mu1, array_mu2, array_mu3))
    print("Дистрибутивності", distributivity(array_mu1, array_mu2, array_mu3))


array_label: list[int] = [x for x in range(150, 205, 5)]
array_mu1: list[float] = [aprocZ(x, 155, 195) for x in array_label]
array_mu2: list[float] = [aprocS(x, 155, 195) for x in array_label]
print("Вектори")
print("A - низька", array_mu1)
print("B - Висока", array_mu2)
fig, ax = plt.subplots(2)
ax[0].plot(array_label, array_mu1, color='#000000', label='A')
ax[0].legend(loc=2)
ax[1].plot(array_label, array_mu2, color='#000000', label='B')
ax[1].legend(loc=2)
plt.show()
part2(array_mu1, array_mu2)
part3(array_label, array_mu1, array_mu2)
part4(array_mu1, array_mu2)
```