

НУЛП, ІКНІ, САП		Тема	оцінка	підпис
КН-406	1(номер лаб)	МЕТОДИ РОЗВ'ЯЗАННЯ ЗАДАЧ БАГАТОКРИТЕРІАЛЬНОЇ ОПТИМІЗАЦІЇ З ВИКОРИСТАННЯМ АДИТИВНОЇ, МУЛЬТИПЛІКАТИВНОЇ, МАКСИМІННОЇ ТА МІНІМАКСНОЇ ЗГОРТКИ		
Ваврик Р.Р.				
№ залікової:				
Методи багатокритеріальної оптимізації			Викладач: к.т.н., доц. каф. Мельник М.Р.	

Мета: Ознайомитися з основними можливостями методів згортки критеріїв, навчитися будувати комплексний критерій та застосовувати їх до розв'язання задач багатокритеріальної оптимізації.

Завдання:

1. Ознайомитися з основними теоретичними відомостями методів розв'язання задач багатокритеріальної оптимізації з використанням узагальненого (інтегрального) критерію оптимальності.

2. Отримати завдання від викладача.

3. Сформулювати задачу багатокритеріальної оптимізації.

4. Визначити вагові коефіцієнти критеріїв оптимальності.

5. Розв'язати сформульовану задачу багатокритеріальної оптимізації використовуючи адитивну, мультиплікативну максимінну та мінімаксну згортки.

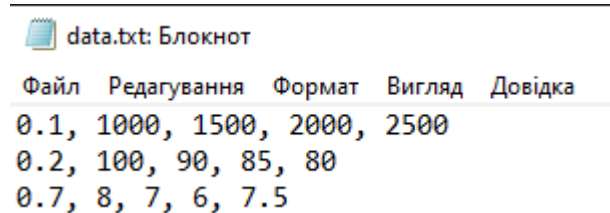
6. Провести аналіз отриманих результатів та оформити звіт.

Деяка фірма планує організувати виробництво дверей. На даний момент існує можливість придбати 4 варіанти станків. Параметри станків, які використовують для виготовлення дверей наведено в таблиці 1. Використовуючи адитивну, мультиплікативну, MaxMin та MinMax згортки – визначити найоптимальніший варіант.

	а	Станок 1	Станок 2	Станок 3	Станок 4
Продуктивність	0.4	1000	1500	2000	2500
Надійність	0.2	100	90	85	80
Вартість	0.4	8	7	6	7.5

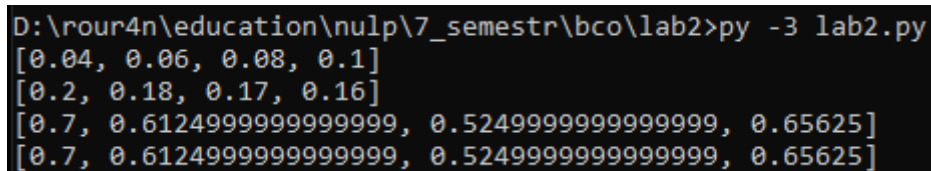
Виконання роботи:

Для початку я записав вхідні дані у текстовий документ та. Після цього я зчитав дані з файлу та перемножив кожен з критеріїв на ваговий коефіцієнт та поділив на максимальне значення.



Файл	Редагування	Формат	Вигляд	Довідка
0.1, 1000, 1500, 2000, 2500				
0.2, 100, 90, 85, 80				
0.7, 8, 7, 6, 7.5				

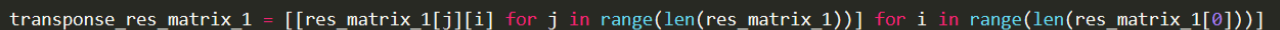
Рис. 1. Вхідні дані



```
D:\rour4n\education\nulp\7_semestr\bco\lab2>py -3 lab2.py
[0.04, 0.06, 0.08, 0.1]
[0.2, 0.18, 0.17, 0.16]
[0.7, 0.6124999999999999, 0.5249999999999999, 0.65625]
[0.7, 0.6124999999999999, 0.5249999999999999, 0.65625]
```

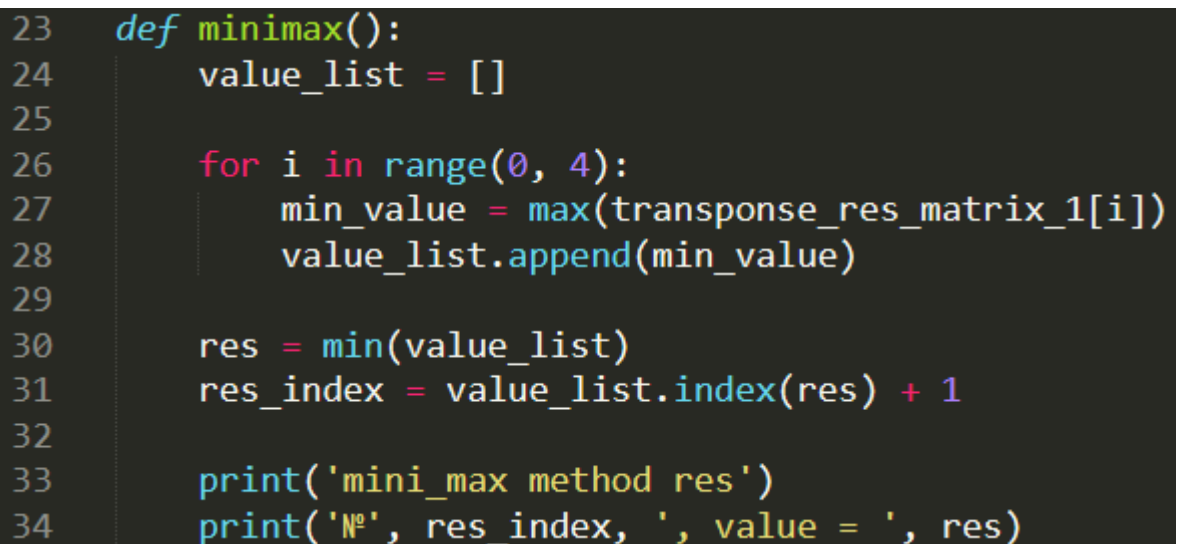
Рис. 2. Значення КО

Для максимінного методу, необхідно спочатку знайти мінімальні значення усіх стовпців, і після цього вибрати максимальне значення. Для мінімаксного навпаки ж – спочатку вибрати максимальні для стовпців значення і вже серед них максимальні. Для легших обрахунків я транспонував матрицю значень КО.



```
transpose_res_matrix_1 = [[res_matrix_1[j][i] for j in range(len(res_matrix_1))] for i in range(len(res_matrix_1[0]))]
```

Рис. 3. Транспонування матриці.



```
23 def minimax():
24     value_list = []
25
26     for i in range(0, 4):
27         min_value = max(transpose_res_matrix_1[i])
28         value_list.append(min_value)
29
30     res = min(value_list)
31     res_index = value_list.index(res) + 1
32
33     print('mini_max method res')
34     print('№', res_index, ', value = ', res)
```

Рис. 4. Код для мінімаксного методу.

```
36 def max_min():
37     value_list = []
38
39     for i in range(0, 4):
40         min_value = min(transpose_res_matrix_1[i])
41         value_list.append(min_value)
42
43     res = max(value_list)
44     res_index = value_list.index(res) + 1
45
46     print('max_min method res')
47     print('№', res_index, ', value = ', res)
```

Рис. 5. Код для максимінного методу.

```
D:\rour4n\education\nulp\7_semestr\bco\lab2>py -3 lab2.py
mini_max method res
№ 3 , value = 0.5249999999999999
max_min method res
№ 4 , value = 0.1
```

Рис. 6. Результати для максимінного та мінімаксного методів.

Для розв'язання задачі адитивним методом необхідно додати критерії помножені на їхню вагу, та знайти максимальний серед них. Мультиплікативний метод аналогічний попередньому, лише потрібно замінити дію додавання множенням.

```
49 def adaptive():
50     all_values = []
51
52     for i in range(0, 4):
53         value = sum(transpose_res_matrix_1[i])
54         all_values.append(value)
55
56     for i in range(0, 4):
57         print(f'Stanok {i + 1} = ', all_values[i])
58
59     res = max(all_values)
60     res_index = all_values.index(res) + 1
61
62     print('adaptive method res')
63     print('№', res_index, ', value = ', res)
```

Рис. 7. Код для адаптивного методу.

```
65 def multiply():
66     all_values = []
67
68     for i in range(0, 4):
69         res = 1
70         for j in transpose_res_matrix_1[i]:
71             res *= j
72         all_values.append(res)
73     print(all_values)
74
75     for i in range(0, 4):
76         print(f'Stanok {i + 1} = ', all_values[i])
77
78     res = max(all_values)
79     res_index = all_values.index(res) + 1
80
81     print('multiply method res')
82     print('№', res_index, ', value = ', res)
```

Рис. 8. Код для мультиплікативного методу.

```
D:\rour4n\education\nulp\7_semestr\bco\lab2>py -3 lab2.py
[0.0056, 0.006614999999999985, 0.00714, 0.0105]
Stanok 1 = 0.0056
Stanok 2 = 0.006614999999999985
Stanok 3 = 0.00714
Stanok 4 = 0.0105
multiply method res
№ 4 , value = 0.0105
Stanok 1 = 0.94
Stanok 2 = 0.8524999999999999
Stanok 3 = 0.7749999999999999
Stanok 4 = 0.91625
adaptive method res
№ 1 , value = 0.94
```

Рис. 9. Результати для максимінного та мінімаксного методів.

Висновки: Для максимінного та мінімаксного критерію я використав такий алгоритм: обирається або найкраща з найнайгірших стратегій, або найгірша з найкращих стратегій. Інколи вони можуть співпадати. Головним недоліком узагальненого адаптивного методу є те, що вагові коефіцієнти призначає сам проектувальник, тобто присутній фактор суб'єктивності. Різні проектувальники можуть призначати абсолютно різні вагові коефіцієнти. Мультиплікативний метод менш чутливий до неточності визначення вагових коефіцієнтів.