

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
КРИВОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ  
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**З В І Т  
Лабораторна робота №1(М)  
«Лінійне програмування»**

**з дисципліни  
«Математичне моделювання»**

Виконавець:

студент групи КІ-22м

Косей М.П.

Керівник:

викладач

Вдовиченко І. Н.

2023

# Лабораторна робота №1(М)

**Тема: Лінійне програмування**

**Мета: Вирішити задачі**

## ЗАВДАННЯ:

### 1) Ознайомитись з теоретичними відомостями до лабораторної роботи.

Математичне програмування - це метод вирішення оптимізаційних задач з використанням математичних моделей та алгоритмів.

Лінійне програмування є одним з методів математичного програмування та використовується для оптимізації лінійних функцій від змінних з обмеженнями в формі лінійних рівнянь або нерівностей.

При розв'язуванні задачі лінійного програмування необхідно побудувати математичну модель, яка описує цільову функцію та обмеження

### 2) Задача 1

У майстерні "ПромАрт" освоєно виробництво столів та тумбочок для торгової мережі. Для їх виготовлення є два види деревини: I - 72 м<sup>3</sup> та II - 56 м<sup>3</sup>. Кожен виріб вимагає обох видів деревини в м<sup>3</sup>:

№	Найменування	I	II
1	Стіл	0,18	0,08
2	Тумбочка	0,09	0,28

З виробництва одного столу "ПромАрт" отримує чистого доходу **1,1 гривні**, а від виробництва однієї тумбочки - **70 копійок**.

Визначте, скільки столів та тумбочок повинна виготовити майстерня з наявної деревини, щоб забезпечити найбільший дохід.

### Вирішення

Для задачі про виробництво столів та тумбочок з обмеженими запасами деревини можна побудувати наступну математичну модель лінійного програмування:

#### Змінні:

$x_1$  - кількість столів, що будуть вироблені

$x_2$  - кількість тумбочок, що будуть вироблені

#### Цільова функція:

$$\text{maximize } 1.11 \times x_1 + 0.7 \times x_2$$

#### Обмеження:

$$0.18 \times x_1 + 0.09 \times x_2 \leq 72 \text{ (обмеження на деревину I)}$$

$$0.08 \times x_1 + 0.28 \times x_2 \leq 56 \text{ (обмеження на деревину II)}$$

$$x_1, x_2 \geq 0 \text{ (неможливо виробити від'ємну кількість виробів)}$$

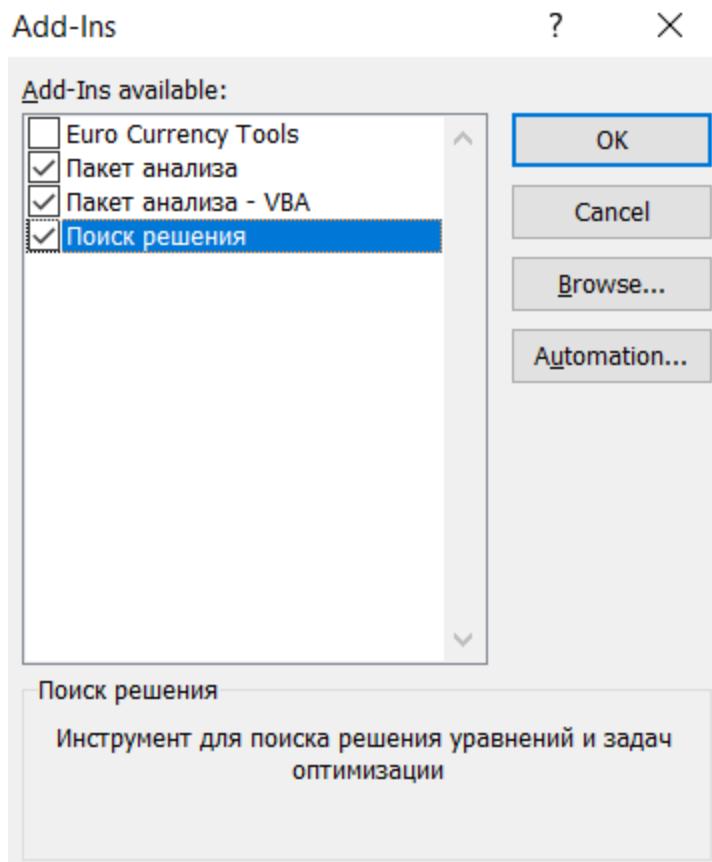
$$x_1, x_2 - \text{цілі числа} \text{ (неможливо виробити дрібну кількість виробів)}$$

Ця модель має ціль максимізувати прибуток, який залежить від кількості

вироблених столів та тумбочок, при тому, що обсяг використання кожного типу деревини не може перевищувати обмежень.

Задача може бути вирішена за допомогою спеціального програмного забезпечення, такого як «MS Excel», яке має вбудовані інструменти розв'язування задач лінійного програмування.

Вмикаємо додаток для вирішення задачи



Вносимо вхідні дані.

	A	B	C	D	E	F	G	H
1	ПромАрт							
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								
13								

Дохід від виробництва одиниці продукції

	Стіл	Тумбочка
1.1	0.7	

Потріби на виготовлення виробів

	Стіл	Тумбочка			
Деревина I	0.18	0.09			
Деревина II	0.08	0.28			

Витрати сировини

=C12*C7 + D12*D7	≤	72			
=C12*C8 + D8*D12	≤	56			

Запаси сировини

Максимальний дохід

=C12*C4+D12*D4			

## Запускаємо вирішувач.

The screenshot shows the Microsoft Excel ribbon with the "Data" tab selected. In the "Data" tab, the "Analysis" group is open, showing options like "Text to Columns", "Duplicates", "Validation", "What-if Analysis", "Group", "Ungroup", "Subtotal", "Outline", and "Show Detail". The "Solver" icon is highlighted. A tooltip for "Solver" is visible, stating: "Средство анализа \"что если\" определяет оптимальное значение целевой ячейки, изменяя значения ячеек, которые используются для расчета значения целевой ячейки." Below the ribbon, a worksheet titled "Laba1(M).xism" is displayed with columns A through E and rows 1 through 3. Row 1 contains the text "ПромАрт". Row 2 is empty. Row 3 contains the text "Стіл" in column C and "Тумбочка" in column D. The "Solver Parameters" dialog box is open over the worksheet. The "Objective" section has "\$H\$12" selected as the target cell, with "Maximise" selected. The "By changing cells" section has "\$C\$12:\$D\$12" selected. The "Subject to the constraints" section lists several linear inequalities: \$C\$12 = целое, \$C\$12 >= 0, \$D\$12 = целое, \$D\$12 >= 0, \$F\$7 <= \$H\$7, and \$F\$8 <= \$H\$8. To the right of these constraints are buttons for "Добавить" (Add), "Изменить" (Edit), "Удалить" (Delete), "Сбросить" (Clear), and "Загрузить/сохранить" (Load/Save). The "Set Objective" section has a checked checkbox "Сделать переменные без ограничений неотрицательными" (Make variables non-negative). The "Select a Solving Method" section has "Поиск решения линейных задач симплекс-методом" (Simplex method for linear programming problems) selected. The "Help" section at the bottom left says "Справка" (Help), the "Find Solution" button is highlighted with a blue border, and the "Close" button is also present.

File Home Insert Page Layout Formulas Data Review View Developer Office Tab Foxit PDF Terabox

From Access From Web From Text Sources Existing Connections Refresh All Properties Edit Links Connections Sort Filter Advanced Sort & Filter Text to Columns Duplicates Validation Consolidate What-if Analysis Group Ungroup Subtotal Show Detail Hide Detail Анализ

Laba1(M).xism – Microsoft Excel (Product Activation Failed)

H11 Максимальний дохід

Лаба1(M).xism

1 ПромАрт

2

3 Стіл Тумбочка

Параметри пошуку розв'язання

Оптимизувати цільову функцію:

До:  Максимум  Мінімум  Значення:

Ізмінює ячейки перемінних:

В згідності з обмеженнями:

\$C\$12 = ціле  
\$C\$12 >= 0  
\$D\$12 = ціле  
\$D\$12 >= 0  
\$F\$7 <= \$H\$7  
\$F\$8 <= \$H\$8

Добавить

Изменить

Удалить

Сбросить

Загрузить/сохранить

Сделать переменные без ограничений неотрицательными

Выберите метод решения: Поиск решения лінійних задач симплекс-методом Параметри

Метод розв'язання

Для гладких нелінійних задач используйте пошук розв'язання нелінійних задач методом ОПГ, для лінійних задач - пошук розв'язання лінійних задач симплекс-методом, а для негладких задач - еволюційний пошук розв'язання.

Справка Найти решение Закрыть

Результати вирішення

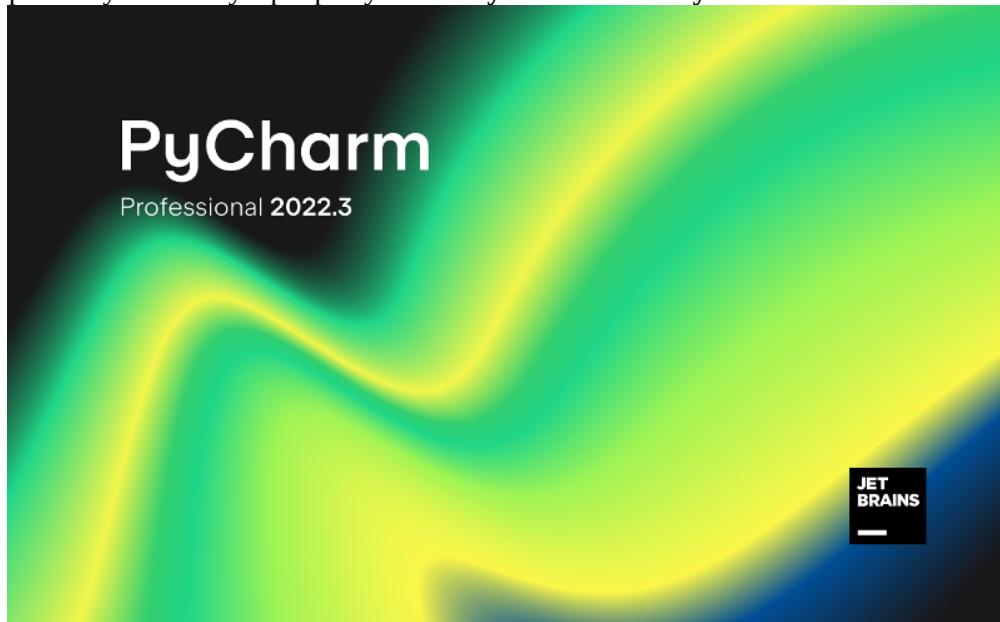
	A	B	C	D	E	F	G	H
1	ПромАрт							
2								
3			Стіл	Тумбочка				
4	Дохід від виробництва одиниці продукції		1.1	0.7				
5								
6	Потреби на виготовлення виробів	Стіл	Тумбочка		Витрати сировини		Запаси сировини	
7	Деревина I	0.18	0.09		72	$\leq$	72	
8	Деревина II	0.08	0.28		56	$\leq$	56	
9								
10								
11	Обсяг виробництва	Стіл	Тумбочка				Максимальний дохід	
12		350	100				455	
13								

**Графічне вирішення задачі.**

Для вирішення задачі потрібно побудувати графіки функцій:

- цільова функція  $1.11 \times x_1 + 0.7 \times x_2$
  - та обмежень
- $0.18 \times x_1 + 0.09 \times x_2 \leq 72$  (обмеження на деревину I)  
 $0.08 \times x_1 + 0.28 \times x_2 \leq 56$  (обмеження на деревину II)

Використовуємо мову програмування **Python** та **IDE PyCharm**



```

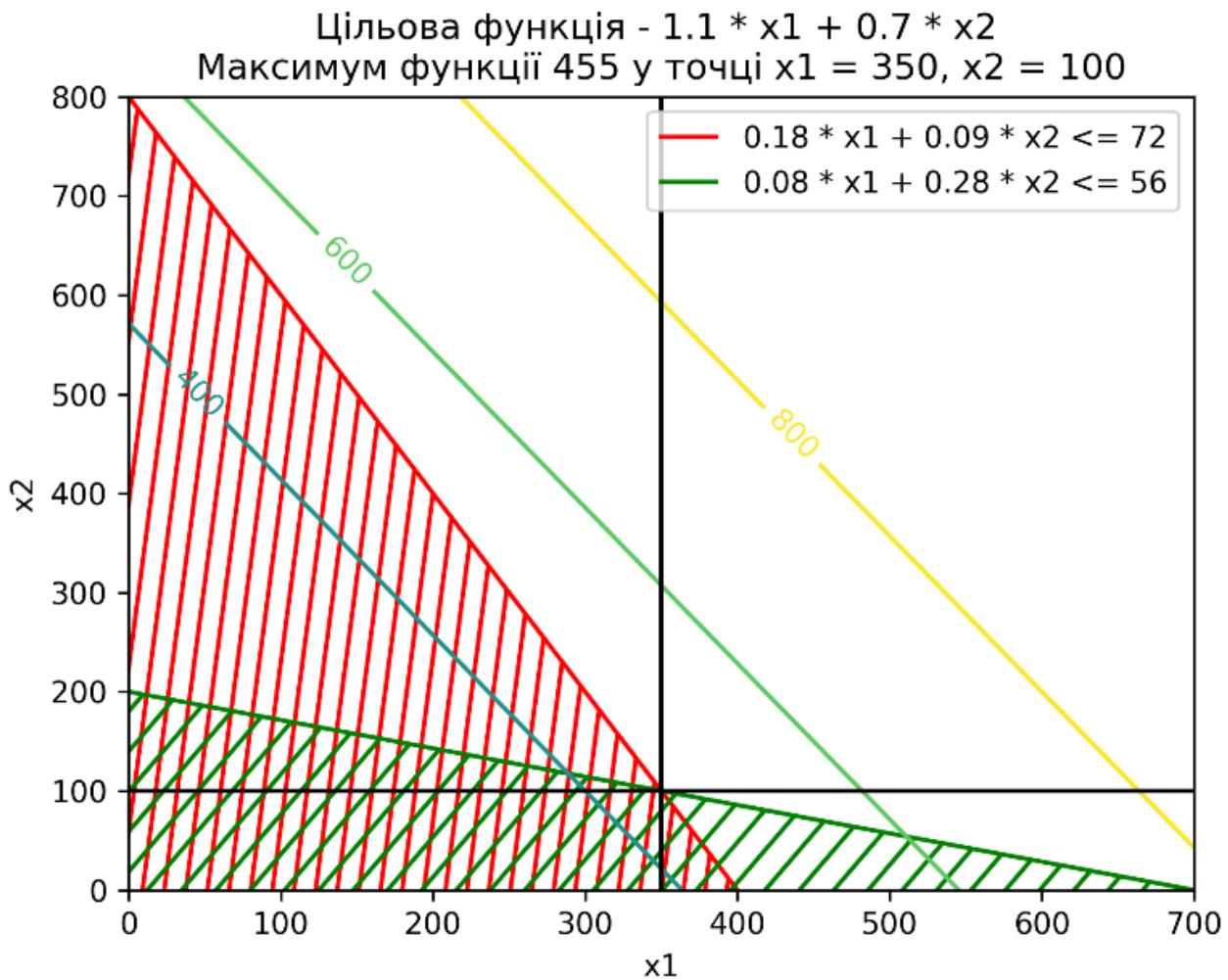
main.py x
1 import matplotlib.pyplot as plt
2 import matplotlib.axes as axes
3 import matplotlib.figure as figure
4 from matplotlib import patheffects
5 import numpy as np
6
7
8 # Визначення функцій
9 2 usages
10 def target_func(x1, x2):
11     return 1.1 * x1 + 0.7 * x2
12
13 1 usage
14 def constraint1(x1, x2):
15     return 72 - 0.18 * x1 - 0.09 * x2
16
17 1 usage
18 def constraint2(x1, x2):
19     return 56 - 0.08 * x1 - 0.28 * x2
20

```

```

21 # Побудова графіків
22 fig, ax = plt.subplots() # type:figure.Figure, axes.Axes
23
24 x1 = np.linspace(0, 700, 701)
25 x2 = np.linspace(0, 800, 801)
26 X1, X2 = np.meshgrid(x1, x2)
27
28 # Обмеження
29 Z1 = constraint1(X1, X2)
30 Z2 = constraint2(X1, X2)
31
32 C_Z1=ax.contour(X1, X2, Z1, levels=[0], colors='r')
33 C_Z2=ax.contour(X1, X2, Z2, levels=[0], colors='g')
34 plt.setp(C_Z2.collections, path_effects=[patheffects.withTickedStroke(angle=60, length=30)])
35 plt.setp(C_Z1.collections, path_effects=[patheffects.withTickedStroke(angle=135, length=30)])
36 h1,l1 = C_Z1.legend_elements()
37 h2,l2 = C_Z2.legend_elements()
38 plt.legend([h1[0], h2[0]], ['0.18 * x1 + 0.09 * x2 <= 72', '0.08 * x1 + 0.28 * x2 <= 56'])
39
40 ax.set_xlabel('x1')
41 ax.set_ylabel('x2')
42
43 # Цільова функція
44 z = target_func(X1, X2)
45 CS_Z = ax.contourf(X1, X2, z, levels=[0, 400, 600, 800])
46 ax.clabel(CS_Z, inline=True, fontsize=10)
47
48 #Максимум цільової функції у точці перетину '0.18 * x1 + 0.09 * x2 = 72 та '0.08 * x1 + 0.28 * x2 = 56
49 A = np.array([[0.18, 0.09],
50               [0.08, 0.28]])
51 y = np.array([72, 56])
52
53 x = np.linalg.solve(A, y)
54 S_x1 = x[0]
55 S_x2 = x[1]
56
57 ax.axvline(x=S_x1, color='black')
58 ax.axhline(y=S_x2, color='black')
59
60 ax.set_title('Цільова функція - 1.1 * x1 + 0.7 * x2 \n' +
61             'Максимум функції ' + str(round(target_func(S_x1, S_x2))) + ' у точці x1 = ' + str(round(S_x1)) +
62             ', x2 = ' + str(round(S_x2)))
63
64 plt.savefig('figure.png', dpi=300)

```



### 3) Задача 2

Є три екскаватори різних марок. З їх допомогою треба виконати три види земляних робіт **загальним** обсягом 20 000 м<sup>3</sup>. Час роботи екскаваторів одинаковий, продуктивність у м<sup>3</sup>-годину по кожному виду робіт наведена в таблиці:

Екскаватор	Вид роботи		
	A	B	C
I	105	56	56
II	107	66	83
III	64	38	53

Необхідно розподілити час роботи так, щоб завдання було виконано за найкоротший час.

### Вирішення

Для розв'язання цієї задачі ми можемо скористатися методом лінійного програмування. Давайте визначимо змінні:

Визначимо змінні:

$x_1$  - час, який витрачає екскаватор I на вид робіт A

- x<sub>2</sub>** - час, який витрачає екскаватор II на вид робіт А
- x<sub>3</sub>** - час, який витрачає екскаватор III на вид робіт А
- y<sub>1</sub>** - час, який витрачає екскаватор I на вид робіт В
- y<sub>2</sub>** - час, який витрачає екскаватор II на вид робіт В
- y<sub>3</sub>** - час, який витрачає екскаватор III на вид робіт В
- z<sub>1</sub>** - час, який витрачає екскаватор I на вид робіт С
- z<sub>2</sub>** - час, який витрачає екскаватор II на вид робіт С
- z<sub>3</sub>** - час, який витрачає екскаватор III на вид робіт С

## Цільова функція:

$$\text{minimize } x_1 + x_2 + x_3 + y_1 + y_2 + y_3 + z_1 + z_2 + z_3$$

## **Обмеження:**

**105 × x<sub>1</sub> + 107 × y<sub>1</sub> + 64 × z<sub>1</sub> + (обсяг робіт А)**

**56 × x<sub>2</sub> + 107 × y<sub>2</sub> + 107 × z<sub>2</sub> + (обсяг робіт В)**

$$56 \times x_3 + 83 \times y_3 + 53 \times z_3 = \text{(обсяг робіт С)} \\ = 20\ 000 \text{ (обмеження на загальний обсяг робіт)}$$

$x_1 + y_1 + z_1 = x_2 + y_2 + z_2$  (час роботи екскаваторів однаковий)

$x_2 + y_2 + z_2 = x_3 + y_3 + z_3$  (час роботи екскаваторів однаковий)

$x_1, y_1, z_1, x_2, y_2, z_2, x_3, y_3, z_3 \geq 0$  (час роботи не повинен бути від'ємним)

Задача може бути вирішена за допомогою спеціального програмного забезпечення, такого як LibreOffice Calc(аналог «MS Excel»), яке має вбудовані інструменти розв'язування задач лінійного програмування.

## Вносимо вхідні дані

## Запускаємо вирішувач

	A	B	C	AutoCorrect Options...	G	H	I	J	K	L
1				✓ AutoInput ImageMap						
2				Redact						
3				Auto-Redact						
4		Продуктивність екскаватора I	10	A Goal Seek... Solver... Detective Scenarios... Forms Share Spreadsheet... Protect Sheet... Protect Spreadsheet Structure...		Час роботи екскаватора I	0	0	0	=SUM(I4:K4)
5		Продуктивність екскаватора II	10			Час роботи екскаватора II	0	0	0	=SUM(I5:K5)
6		Продуктивність екскаватора III	64	Macros Development Tools		Час роботи екскаватора III	0	0	0	=SUM(I6:K6)
7				Extension Manager... Customize... Options... Ctrl+Alt+E						
8		Екскаватор I	Екскаватор II	Екскаватор III	Сумарний час					
9	Обсяг робіт А	=C4*I4	=C5*I5	=C6*I6	=SUM(C9:E9)	=	20000			
10	Обсяг робіт В	=D4*I4	=D5*I5	=D6*I6	=SUM(C10:E10)	=	20000			
11	Обсяг робіт С	=E4*K4	=E5*K5	=E6*K6	=SUM(C11:E11)	=	20000			
12										
13										
14							Мінімальний сумарний час роботи			
15							=SUM(I4:K6)			

**Solver**

Target cell:

Optimize result to:

Maximum

Minimum

Value of:

By changing cells:

**Limiting Conditions**

Cell reference	Operator	Value
\$F\$12	=	\$H\$12
\$L\$4	=	\$L\$5
\$L\$5	=	\$L\$6
	<=	

**Options**

Solver engine: LibreOffice Linear Solver

Settings:

- Assume variables as integer
- Assume variables as non-negative
- Epsilon level (0-3): 0
- Limit branch-and-bound depth
- Solving time limit (seconds): 100

## Результати вирішення

## ВИСНОВКИ

В результаті виконаної лабораторної роботи було розглянуто один з методів математичного програмування - лінійне програмування, а також вирішенні задачі оптимізації.

Усі матеріали викладені у репозіторії GitHub, за посиланням <https://github.com/Max11mus/Mathematical-Modeling-Lenear-Programming-Lab1.git>.