МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМ. ІГОРЯ СІКОРСЬКОГО»
НАУКОВО-НАВЧАЛЬНИЙ КОМПЛЕКС
«ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ»
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

# КУРСОВА РОБОТА

З курсу «Теорія прийняття рішень»

Тема: «Розподіл обмежених ресурсів між темами»

Науковий керівник:
Зайченко Ю. П.

Виконав:
Орловський Артем
Студент 4 курсу
групи КА-87

Захищено з оцінкою

_____

Київ 2021

## РЕФЕРАТ

Курсова робота містить 31 сторінку, 5 таблиць, 1 рисунок. Було використано 2 джерела.

Ключові слова: прийняття рішень, оптимальний, асигнування, розподіл.

Мета роботи: дослідити задачі оптимального розподілу асигнування на виконання різних видів робіт з метою максимізації прибутку.

Побудувати відповідну математичну модель. Написати програмний продукт, що дозволяє вирішувати такі завдання при різних вхідних даних.

## ABSTRACT

The course work contains 31 pages, 5 tables, 1 drawing. 2 sources were used.

Keywords: decision making, optimal, allocation, distribution.

Purpose: To investigate the problems of optimal allocation of allocations to perform different types of work in order to maximize profit.

Construct an appropriate mathematical model (dynamic programming problem). Write a software product that allows you to solve such problems with different input data.

**ЗМІСТ**

# ВСТУП

Сьогодні технології займають провідну частину повсякденного життя. Вони стрімко розвиваються, люди знаходять нове застосування, виникають нові потреби. Багато сфер виробництва стикаються з передовими технологіями кожного дня і необхідно розуміти, що варто впроваджувати, на чому акцентувати увагу на підприємстві. Саме тому прийняття належних рішень відіграє важливу роль у суспільстві, і, як наслідок, важливою є дисципліна, що складає теорію для даних задач.

Багато підприємств в пошуку кращого можливого прибутку стикається з різними проблемами. Ресурси обмежені, необхідно розуміти яким відділам скільки ресурсів необхідно виділяти і для яких видів робіт для максимізації кінцевих доходів. Через це виникає потреба у вирішенні задачі розподілу ресурсів, зокрема один з її варіантів розподіл ресурсів за темами для різних видів робіт.

Завдяки цьому вирішення цієї задачі для кожного підприємства є нагально актуальним. З такою задачею може стикатися будь-яка організація або фізична особа в будь-який час.

## МАТЕМАТИЧА ПОСТАНОВКА ЗАДАЧІ

Керівництво НВО «Пластмаш», до складу якого входять науково - дослідні відділи, розподіляє асигнування на виконання НДР і ДКР (науково - дослідних і дослідно - конструкторських робіт).

Кожний з $S$ відділів представляє керівництву НВО дані трьох видів. Інформація першої групи відноситься до проведення пошукових досліджень не визначеного характеру. Якщо на дослідження такого роду в відділі $j$ виділяється $v_j$ тис грн, то оцінка очікуваного довгострокового доходу дорівнює $P_j(v_j)$ млн грн. Інформація другої групи відноситься до продукції, по якій пошукові дослідження вже завершені, і для випуску якої потрібні ДКР і випробування. Для таких проектів асигнування в обсязі $w_j$ тис грн, згідно з наявною оцінкою, дадуть в кінцевому рахунку дохід у розмірі $Q_j(w_j)$ млн грн.

До третьої групи відноситься інформація, яка пов'язана з поліпшенням якості випущеної продукції. Витрати $x_j$ тис грн повинні, згідно з оцінками, принести $R_j(x_j)$ млн грн додаткового доходу.

Загальна сума асигнувань на всі проекти НДР і ДКР становить $N$ тис грн, Верхня межа асигнувань, які може отримати відділ $j$ – $L_j$ тис грн.

Потрібно розподілити асигнування між відділами так, щоб забезпечити максимізацію загального доходу НВО від проведення всіх НДР і ДКР.

$S = 4$, $N = 100$, $L_j = 30$, $j = 1, 2, 3, 4$.

Таблиця 1.1

j = 1.

| v | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P_1(v)$ | 0 | 8 | 16 | 23 | 30 | 36 | 41 | 45 | 49 | 53 | 55 | 57 | 59 | 60 | 61 | 62 |
| v | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | |
| $P_1(v)$ | 63 | 64 | 65 | 65 | 65 | 65 | 66 | 66 | 67 | 67 | 67 | 67 | 67 | 67 | 67 | |
| w | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| $Q_1(w$ | 0 | 10 | 20 | 29 | 38 | 46 | 54 | 61 | 67 | 72 | 77 | 79 | 82 | 84 | 86 | 86 |

| w | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $Q_1(w$ | 87 | 88 | 89 | 89 | 90 | 90 | 90 | 90 | 90 | 90 | 90 | 90 | 90 | 90 | 90 | 90 |
| x | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| $BR_1(x$ | 0 | 15 | 29 | 42 | 54 | 65 | 75 | 84 | 92 | 100 | 107 | 113 | 118 | 122 | 125 | 125 |
| x | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | |
| $R_1(x)$ | 127 | 128 | 128 | 128 | 128 | 128 | 128 | 128 | 128 | 128 | 128 | 128 | 128 | 128 | 128 | |

Таблиця 1.2

j = 2.

| v | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P_2(v)$ | 0 | 20 | 39 | 56 | 72 | 87 | 100 | 112 | 123 | 134 | 144 | 153 | 162 | 170 | 177 | 183 |
| v | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | |
| $P_2(v)$ | 188 | 192 | 195 | 197 | 198 | 199 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | |
| w | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| $Q_2(w$ | 0 | 15 | 28 | 40 | 51 | 61 | 70 | 79 | 87 | 93 | 98 | 102 | 105 | 107 | 108 | 109 |
| w | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | |
| $Q_2(w$ | 110 | 110 | 110 | 110 | 110 | 110 | 110 | 110 | 110 | 110 | 110 | 110 | 110 | 110 | 110 | |
| x | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| $R_2(x)$ | 0 | 10 | 10 | 29 | 38 | 46 | 51 | 55 | 58 | 60 | 70 | 79 | 87 | 93 | 100 | 103 |
| x | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | |
| $R_2(x)$ | 105 | 107 | 109 | 112 | 115 | 117 | 118 | 119 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | |

Таблиця 1.3

j = 3.

| v | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P_3(v)$ | 0 | 13 | 25 | 36 | 46 | 55 | 64 | 72 | 79 | 85 | 90 | 94 | 97 | 99 | 100 | 100 |
| v | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | |
| $P_3(v)$ | 101 | 101 | 102 | 102 | 102 | 103 | 103 | 104 | 104 | 104 | 105 | 105 | 105 | 105 | 105 | |
| w | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

| $Q_3(w$ | 0 | 10 | 20 | 28 | 36 | 43 | 50 | 56 | 61 | 65 | 67 | 69 | 70 | 70 | 70 | 70 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| w | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | |
| $Q_3(w$ | 70 | 70 | 70 | 70 | 70 | 70 | 70 | 70 | 70 | 70 | 70 | 70 | 70 | 70 | 70 | |
| x | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| $R_3(x)$ | 0 | 15 | 29 | 42 | 53 | 63 | 72 | 80 | 86 | 91 | 95 | 98 | 100 | 100 | 100 | |
| x | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | |
| $R_3(x)$ | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | |

Таблиця 1.4

j = 4.

| v | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P_4(v)$ | 0 | 8 | 16 | 23 | 30 | 36 | 41 | 45 | 49 | 53 | 55 | 57 | 59 | 60 | 61 | 62 |
| v | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | |
| $P_4(v)$ | 63 | 64 | 65 | 65 | 65 | 65 | 66 | 66 | 67 | 67 | 67 | 67 | 67 | 67 | 67 | |
| w | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| $Q_4(w$ | 0 | 20 | 39 | 56 | 72 | 87 | 100 | 112 | 123 | 134 | 144 | 153 | 162 | 170 | 177 | 183 |
| w | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | |
| $Q_4(w$ | 188 | 192 | 195 | 197 | 198 | 199 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | |
| x | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| $R_4(x)$ | 0 | 10 | 20 | 29 | 38 | 46 | 51 | 55 | 58 | 60 | 70 | 79 | 87 | 93 | 100 | 105 |
| x | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | |
| $R_4(x)$ | 109 | 112 | 115 | 117 | 118 | 119 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | |

## МАТЕМАТИЧНА МОДЕЛЬ

Математична модель даної задачі має даний вигляд:

$$\max(z) = \sum_{j=1}^{S=4} f(y_j) = \max \sum_{j=1}^{S=4} (\mathrm{P}_j(\mathrm{V}_j) + \mathrm{Q}_j(\mathrm{W}_j) + \mathrm{R}_j(\mathrm{X}_j))$$

при обмеженнях:

$$\sum_{j=1}^{4} (V_j + W_j + \mathrm{X}_j) \leq \mathrm{N}$$

$$V_j + W_j + \mathrm{X}_j \leq \mathrm{L}_j \ , \ j = 1..4$$

$$V_j, W_j, \mathrm{X}_j \geq 0$$

## АНАЛІТИЧНИЙ РОЗВ'ЯЗОК

Дана задача являє собою модифікацію задачі розподілу ресурсів, тому потрібно спочатку згадати задачу в простішому випадку.

Нехай дано наступну задачу:

$$z = \sum_{j=1}^{n} f_j(x_j) \to max$$

При обмеженнях:

$$\sum_{j=1}^{n} a_j x_j \leq b\,;$$

$$a_j \leq 0; x_j \geq 0, b > 0$$

Цільова функція задачі та обмеження є адитивними функціями. Для того, щоб знайти глобальний максимум, застосуємо метод динамічного програмування. Припустимо, що всі $a_j, b$ – цілі числа а також факт того що всі змінні набувають лише цілочисельних значень.

Позначимо $z^*$ як глобальний максимум $z$ за умови $\sum_{j=1}^{n} a_j x_j \leq b$. Оберемо певне значення $x_n$, і зафіксувавши його, максимізуємо z по іншим змінним $x_1, x_2, \dots, x_{n-1}$. Припустимо, що така максимізація проведена для всіх можливих значень $x_n$. Тоді $z^*$ буде найбільшим з усіх можливих значень $z$. Формально цей процес записується наступним чином:

$$z(x_n) = \max_{x_1, x_2, \dots, x_{n-1}} \left\{ \sum_{j=1}^{n} f_j(x_j) \right\} = f_n(x_n) + \max_{x_1, x_2, \dots, x_{n-1}} \left\{ \sum_{j=1}^{n-1} f_j(x_j) \right\},$$

причому

$$\sum_{j=1}^{n-1} a_j x_j \leq b - a_n x_n$$

Оскільки $\max\limits_{x_1,\dots,x_{n-1}} \sum_{j=1}^{n-1} f_j(x_j)$ для невід'ємних цілих чисел, що задовольняють умові $\sum_{j=1}^{n-1} a_j x_j \le b - a_n x_n$, залежить від $b - a_n x_n$, то позначимо

$$\max_{x_1,\dots,x_{n-1}} \sum_{j=1}^{n-1} f_j(x_j) = \Lambda_{n-1}(b - a_n x_n)$$

Тепер введемо важливе припущення. Нехай ми обчислили $\Lambda_{n-1}(b - a_n x_n)$ для всіх допустимих цілих значень $x_n = \left\{0,1,\dots,\left[\frac{b}{a_n}\right]\right\}$, де $\left[\frac{b}{a_n}\right]$ означає цілу частину від $\frac{b}{a_n}$.

Тоді підставляючи дане значення в формулу знаходження $z(x_n)$, отримаємо

$$z(x_n) = f_n(x_n) + \max_{x_1,x_2,\dots,x_{n-1}} \left\{ \sum_{j=1}^{n-1} f_j(x_j) \right\} = f_n(x_n) + \Lambda_{n-1}(b - a_n x_n)$$

Тоді формула для пошуку $z^*$ буде мати вигляд

$$z^* = \max_{x_n \ge 0} z(x_n) = \max_{x_n \ge 0} \{f_n(x_n) + \Lambda_{n-1}(b - a_n x_n)\}$$

Таким чином, якби було відома функція $\Lambda_{n-1}(b - a_n x_n)$, то фактично дана задача звелася би до задачі з однієї змінною.

Тепер знайдемо $\Lambda_{n-1}(b - a_n x_n)$. Коефіцієнти $a_n$ та $b$ відомі за умовою задачі, по змінній $x_n$ буде проводитися максимізація з відомих значень $\left\{0,1,\dots,\left[\frac{b}{a_n}\right]\right\}$. Тому розглянемо саму функцію $\Lambda_{n-1}(\xi)$. Позначимо її як:

$$\Lambda_{n-1}(\xi) = \max_{x_1,\dots,x_{n-1}} \sum_{j=1}^{n-1} f_j(x_j),$$

за умови:

$$\sum_{j=1}^{n} a_j x_j \le \xi$$

Застосовуючи попередні міркування, що були проведені над функцією $z$, отримаємо:

$$\Lambda_{n-1}(\xi) = \max_{x_{n-1}}\{f_{n-1}(x_{n-1}) + \Lambda_{n-2}(\xi - a_{n-1}x_{n-1})\},$$

де

$$\sum_{j=1}^{n-2} a_j x_j \le \xi - a_{n-1}x_{n-1}$$

Аналогічно обчислюємо $\Lambda_{n-2}(\xi), \Lambda_{n-3}(\xi)$ і т.д. На k-ому кроці отримуємо основне рекурентне співвідношення (ОРС):

$$\Lambda_k(\xi) = \max_{x_k}\{f_k(x_k) + \Lambda_{k-1}(\xi - a_k x_k)\},$$

де

$$\sum_{j=1}^{k} a_j x_j \le \xi$$

Також потрібно розглянути $\Lambda_1(\xi)$:

$$\Lambda_1(\xi) = \max_{0 \le x_1 \le \left[\frac{\xi}{a_1}\right]} f_1(x_1)$$

Отримавши ОРС та $\Lambda_1(\xi)$, організуємо процес обчислень. Спочатку складаємо таблицю динамічного програмування першого кроку (табл. 1) і заповнюємо її результатами обчислень.

Потім на другому кроці (k=2) знаходимо $\Lambda_2(\xi)$ згідно зі співвідношенням:

$$\Lambda_k(\xi) = \max_{0 \le x_2 \le \left[\frac{\xi}{a_2}\right]} \{f_2(x_2) + \Lambda_1(\xi - a_2 x_2)\}$$

Значення $\Lambda_1(\xi - a_2 x_2)$ обираємо із (табл. 3.1)

| $\xi$ | $\Lambda_1(\xi)$ | $x_1^0(\xi)$ |
|---|---|---|
| 0 | | |
| 1 | | |
| … | | |
| b | | |

Таблиця 3.1

І так само результати обчислень для $\Lambda_2$ записуємо у таблицю другого кроку.

Продовжуючи дане міркування і користуючись ОРС послідовно обчислюємо $\Lambda_3(\xi), \Lambda_4(\xi), \ldots \Lambda_k(\xi), \ldots \Lambda_{n-1}(\xi)$.

На $n$ кроці знаходимо

$$z^* = \Lambda_n(\xi = b) = \max_{0 \le x_n \le \left[\frac{b}{a_n}\right]} \{f_n(x_n) + \Lambda_{n-1}(b - a_n x_n)\}$$

І отримуємо відповідне оптимальне значення $z$, а також $x_1, \ldots, x_n$, за яких воно досягається.

Тепер перейдемо до задачі розподілу ресурсів за темами.

Перепишемо для початку математичну модель

$$\max(z) = \sum_{j=1}^{S=4} f(y_j) = \max \sum_{j=1}^{S=4} (\mathrm{P}_j(\mathrm{V}_j) + \mathrm{Q}_j(\mathrm{W}_j) + \mathrm{R}_j(\mathrm{X}_j))$$

при обмеженнях

$$\sum_{j=1}^{4} (V_j + W_j + \mathrm{X}_j) \le \mathrm{N}$$

$$V_j + W_j + \mathrm{X}_j \le \mathrm{L}_j\ ,\ j\ = 1..4$$

$$V_j, W_j, \mathrm{X}_j \ge\ 0$$

Враховуючи дану модель, в загальному випадку необхідно вирішити наступну задачу:

$$\max(z) = \sum_{j=1}^{S}\left(\sum_{i=1}^{M} f_{ij}(y_{ij})\right)$$

при обмеженнях

$$\sum_{j=1}^{S}\left(\sum_{i=1}^{M} y_{ij}\right) \leq \mathrm{N}$$

$$\sum_{i=1}^{M} y_{ij} \leq L_j$$

$$j = 1 \ldots \mathrm{S}, \mathrm{i} = 1 \ldots M$$

$$y_{ij} \geq 0$$

Для конкретно випадку даної задачі варіанту вийде S=4, M=3 та:

$$\sum_{i=1}^{M=3} y_{ij} = V_j + W_j + \mathrm{X}_j$$

$$\sum_{i=1}^{M} f_{ij}(y_{ij}) = \mathrm{P}_j(\mathrm{V}_j) + \mathrm{Q}_j(\mathrm{W}_j) + \mathrm{R}_j(\mathrm{X}_j)$$

де, нехай, $V_j = y_{1j}, W_j = y_{2j}, X_j = y_{3j}$ , та $\mathrm{P}_j(\mathrm{V}_j) = f_{1j}(y_{1j})$, $\mathrm{Q}_j(\mathrm{W}_j) = f_{1j}(y_{1j})$, $\mathrm{R}_j(\mathrm{X}_j) = f_{3j}(y_{3j})$.

Введемо означення $\boldsymbol{\theta_j} = \begin{pmatrix} V_j \\ W_j \\ X_j \end{pmatrix}$, $f_j(\boldsymbol{\theta_j}) = \mathrm{P}_j(\mathrm{V}_j) + \mathrm{Q}_j(\mathrm{W}_j) + \mathrm{R}_j(\mathrm{X}_j)$, а також $\gamma_j = V_j + W_j + \mathrm{X}_j$ для зручності для відповідного вектору $\boldsymbol{\theta_j}$.

Тоді задача набуває вигляду:

$$\max(z) = \sum_{j=1}^{\mathrm{S}} f_j(\boldsymbol{\theta_j})$$

при обмеженнях:

$$\sum_{j=1}^{S} \gamma_j \leq \mathrm{N}$$

$$0 \le \gamma_j \le L_j$$

Застосовуємо міркування попередньої задачі. Позначимо $z^*$ як глобальний максимум $z$ за даних умов. Оберемо певне значення $\boldsymbol{\theta_n}$, зафіксувавши його, максимізуємо $z$ по всім іншим змінним $\boldsymbol{\theta_1}, \boldsymbol{\theta_2}, \dots, \boldsymbol{\theta_{n-1}}$.

$$z(\boldsymbol{\theta_n}) = \max_{\boldsymbol{\theta_1},\boldsymbol{\theta_2},\dots,\boldsymbol{\theta_{n-1}}} \left\{ \sum_{j=1}^{n} f_j(\boldsymbol{\theta_j}) \right\} = f_n(\boldsymbol{\theta_n}) + \max_{\boldsymbol{\theta_1},\boldsymbol{\theta_2},\dots,\boldsymbol{\theta_{n-1}}} \left\{ \sum_{j=1}^{n-1} f_j(\boldsymbol{\theta_j}) \right\},$$

Причому,

$$\sum_{j=1}^{n-1} \gamma_j \le N - \gamma_n$$

Так само позначаємо:

$$\max_{\boldsymbol{\theta_1},\boldsymbol{\theta_2},\dots,\boldsymbol{\theta_{n-1}}} \sum_{j=1}^{n-1} f_j(\boldsymbol{\theta_j}) = \Lambda_{n-1}(N - \gamma_n)$$

Тепер введемо аналогічне припущення. Нехай було обчислено $\Lambda_{n-1}(N - \gamma_n)$ для всіх допустимих цілих значень $\boldsymbol{\theta_n}$. Тоді можна порахувати наступне значення функції:

$$z(\boldsymbol{\theta_n}) = f_n(\boldsymbol{\theta_n}) + \max_{\boldsymbol{\theta_1},\boldsymbol{\theta_2},\dots,\boldsymbol{\theta_{n-1}}} \left\{ \sum_{j=1}^{n-1} f_j(\boldsymbol{\theta_j}) \right\} = f_n(\boldsymbol{\theta_n}) + \Lambda_{n-1}(N - \gamma_n)$$

Та знайти максимум:

$$z^* = \max_{\boldsymbol{\theta_S}, \text{такі що } 0 \le \gamma_S \le \min(\xi, L_S)} \{\phi_S(\boldsymbol{\theta_S}) + \Lambda_{S-1}(\xi - \gamma_S)\}$$

Потрібно пам'ятати, що пошук $\boldsymbol{\theta_S}$ ведеться серед значень які задовольняють умові $0 \le \gamma_S \le \min(\xi, L_S)$.

Таким чином, продовжуючи міркування з попередньої задачі так само можна продовжити ідею рішення, отримавши формули:

$$\mathrm{z}^* = \Lambda_S(\xi = N) = \max_{\boldsymbol{\theta_S}, \text{такі що } 0 \le \gamma_S \le \min(\xi, L_S)} \{f_S(\boldsymbol{\theta_S}) + \Lambda_{S-1}(\xi - \gamma_S)\}$$

За умов

$$\sum_{j=1}^{n} \gamma_j \leq N$$

Також відповідне ОРС:

$$\Lambda_k(\xi) = \max_{\boldsymbol{\theta_k}, \text{такі що } 0 \leq \gamma_k \leq \min(\xi, L_k)} \{f_k(\boldsymbol{\theta_k}) + \Lambda_{k-1}(\xi - \gamma_k)\}$$

За умов

$$\sum_{j=1}^{k} \gamma_j \leq \xi$$

Та відповідна формула для $\Lambda_1$:

$$\Lambda_1(\xi) = \max_{\boldsymbol{\theta_k}, \text{такі що } 0 \leq \gamma_1 \leq \min(\xi, L_1)} f_1(\boldsymbol{\theta_1})$$

За умов

$$\gamma_1 \leq \xi$$

Базовий алгоритм для задачі отриманий, але якщо на цьому зупинитись, то дане рішення буде працювати відносно довго. Всіх допустимі значення $\boldsymbol{\theta_k}$ на $k$ кроці складуть велику кількість можливих значень для перевірки, наприклад якщо взяти всіх допустимі значень $\mathrm{V}_k, \mathrm{W}_k, \mathrm{X}_k$ порахувати для кожного по 30 варіантів значень згідно з варіантом, отримаємо $30^3 = 27000$ можливих значень, від чого сильно зросте складність алгоритму. Тому застосуємо інший підхід.

Нехай при пошуці оптимального варіанту $\boldsymbol{\theta_k}$ зафіксуємо певне число $\gamma_k^*$ серед $[0, \dots, \min(\xi, L_k)]$. Розуміємо, що для певного фіксованого числа $\gamma_k^*$ існують різні варіанти значення $\boldsymbol{\theta_k}$, а оптимальне значення $f_k(\boldsymbol{\theta_k})$ для даного фіксованого $\gamma_k^*$ серед них одне. Тому можна застосувати таку ж саму задачу і для пошуку оптимального $\boldsymbol{\theta_k}$ за певного фіксованого $\gamma_k^*$.

Нехай необхідно вирішити наступну задачу:

$$f_k(\boldsymbol{\theta_k}) \to max$$

За умов

$$0 \le \gamma_k \le \gamma_k^*$$

Якщо переписати $f_k(\boldsymbol{\theta_k})$, $\boldsymbol{\theta_k}$, та $\gamma_k$ в термінах, що були дані в початковій задачі, отримаємо:

$$\max \mathrm{P}_k(\mathrm{V}_k) + \mathrm{Q}_k(\mathrm{W}_k) + \mathrm{R}_k(\mathrm{X}_k)$$

при обмеженнях:

$$V_k + W_k + \mathrm{X}_k \le \gamma_k^* \ , \ j = 1..4$$

$$V_j, W_j, \mathrm{X}_j \ge 0$$

І як бачимо, отримали таку саму підзадачу розподілу ресурсів в задачі розподілу ресурсів між темами. Вирішуємо її методом, що описаний вище на початку до пояснення базової задачі розподілу ресурсів, а отриманий результат від цієї задачі нехай тоді позначимо $f_k(\boldsymbol{\theta_k})^*$ на місці $f_k(\boldsymbol{\theta_k})$ у формулі для $\Lambda_k$. Так рахуємо $f_k(\boldsymbol{\theta_k})^*$ для кожного допустимого $\gamma_k^*$. Відповідно тоді частина ОРС задачі пошуку оптимального розподілу ресурсів за темами може бути переписана в наступному вигляді

$$\Lambda_k(\xi) = \max_{0 \le \gamma_k^* \le \min(\xi, L_k)} \{f_k(\boldsymbol{\theta_k})^* + \Lambda_{k-1}(\xi - \gamma_k^*)\}$$

Для кроку k=1 відповідно

$$\Lambda_1(\xi) = \max_{0 \le \gamma_1^* \le \min(\xi, L_1)} f_1(\boldsymbol{\theta_1^*})$$

Таким чином, отримали схожий алгоритм пошуку глобального максимуму задачі розподілу ресурсів за темами, застосовуючи на ітераціях $k$ базовий алгоритм для задачі розподілу ресурсів.

# АЛГОРИТМ РІШЕННЯ ЗАДАЧІ

Заданий алгоритм буде імплементовано на основі двох основних формул ОРС для задачі розподілу ресурсів між темами та задачі розподілу одного обмеженого ресурсу, що були розглянуті в попередній секції аналітичного розв’язку.

Спочатку розглянемо задачу розподілу одного обмеженого ресурсу. Для розв’язку достатньо ефективно застосувати основне рекурентне співвідношення в прямому вигляді як певну рекурентну функцію всередині алгоритму. Рекурсивно спускаючись по ітераціях $k$ від $n$ до 1 можна знайти оптимальний розв'язок для даної задачі через ідею виклику функції самої себе з іншими аргументами.

Так само розглядаємо алгоритм вирішення розподілу ресурсів між темами. Аналогічно імплементується ОРС з обмеженням в циклі за певним $\gamma_k^*$. В свою чергу, на кожній такій ітерації буде викликатися рекурсивна функція підрахунку оптимального рішення першої задачі розподілу з одним обмеженим ресурсом.

Додаючи структури даних для збереження результатів рекурсій для обох алгоритмів, вводимо засіб мемоізації для кожної функції. У аналітичному розв’язку дана структура була представлена певною таблицею результатів обчислень рекурсивних функцій за певних значень $\xi$, тож використаємо даний спосіб.

Аналізуючи складність алгоритму, оцінка динамічне програмування складає $O(n^2)$ операцій, що значно менше ніж випадок простого перебору $O(2^n)$.

# ОПИС ПРОГРАМНОГО ПРОДУКТУ

Програмний продукт був виконаний мультипарадигменною мовою програмування Python у середовищі розробки VS Code. Застосована бібліотека pandas для роботи з таблицями формату Microsoft Excel для перетворення вхідних даних з таблиці у необхідний формат для роботи алгоритму, numpy для ефективної роботи арифметичних операцій над числовими масивами та середовище Jupyter Notebook, яке відображає графічний інтерфейс для користувача та виконує скрипти у покрокових «клітинках».

Файл скрипта та користувацького інтерфейсу виглядає наступним чином

```
1 import numpy as np
2 import pandas as pd
3 from collections import defaultdict
```

```
1 from solution import ThemeSolver
2 from utils import get_report, drop_missing, write_report_to_file, parse_data
```

```
1 %load_ext autoreload
2 %autoreload 2
```

```
1 profits_list, investments_list = parse_data("data/data.xlsx")
```

```
1 L1 = int(input("Введіть L1: "))
2 L2 = int(input("Введіть L2: "))
3 L3 = int(input("Введіть L3: "))
4 L4 = int(input("Введіть L4: "))
5
6 N = int(input("Введіть N: "))
7 L = [L1,L2,L3,L4]
```

```
Введіть L1: 30
Введіть L2: 30
Введіть L3: 30
Введіть L4: 30
Введіть N: 100
```

У клітинці 5 запропонують ввести дані обмежень верхніх меж асигнувань відділів та виділеного бюджету в поля підсвічені чорним прямокутником.

```
L1 = int(input("Введіть L1: "))
L2 = int(input("Введіть L2: "))
L3 = int(input("Введіть L3: "))
L4 = int(input("Введіть L4: "))

N = int(input("Введіть N: "))
L = [L1,L2,L3,L4]
```

```
Введіть L1: 30
Введіть L2: 30
Введіть L3: 30

Введіть L4: 30
```

Після цього програма знайде рішення та покаже звіт згідно з яким необхідно інвестувати, та збереже його у файл “report.txt”.

In [6]:
```
solver = ThemeSolver(profits_list, investments_list, L, N)
```

In [7]:
```
%%time
solution = solver.solve()
```

```
Wall time: 8.88 s
```

In [8]:
```
report = get_report(profits_list, profits_list, L, N, solution)
```

In [9]:
```
print(report)
```

```
Загальна сума асигнувань на всі проекти = 100
Верхні межі асигнувань відділів: L_1=30 L_2=30 L_3=30 L_4=30
Максимальний результат = 1018

V=5 W=8 X=11 | Sum=24
V=15 W=9 X=5 | Sum=29
V=9 W=7 X=8 | Sum=24
V=4 W=14 X=5 | Sum=23
Сума всіх інвестицій = 100

P(5) = 36    Q(8) = 67    R(11) = 113    | Sum=216
P(15) = 183    Q(9) = 93    R(5) = 46    | Sum=322
P(9) = 85    Q(7) = 56    R(8) = 86    | Sum=227
P(4) = 30    Q(14) = 177    R(5) = 46    | Sum=253
Сума всіх доходів = 1018
```

In [10]:
```
write_report_to_file(report,"report.txt")
```

## АНАЛІЗ РЕЗУЛЬТАТІВ

Після виконання, програма повертає результат у вигляді пари максимуму функції $z$ та структури даних, що зберігає аргументи, за яких цей максимум був досягнутий в зручній формі. Отримуємо наступний результат:

Максимальний результат = 1018

Асигнування:

$$V_1 = 5, W_1 = 8, X_1 = 11$$

$$V_2 = 15, W_2 = 9, X_2 = 5$$

$$V_3 = 9, W_3 = 7, X_3 = 8$$

$$V_4 = 4, W_4 = 14, X_4 = 5$$

Виконаємо перевірку обмежень:

Відділ 1: 5+8+11= 24

Відділ 2: 15+9+5 = 29

Відділ 3: 9+7+8 = 24

Відділ 4:4+14+5 = 23

(5+8+11)+(15+9+5)+(9+7+8)+(4+14+5)= 100

Також виконаємо перевірку доходів згідно таблиці:

36+67+113 = 216

183+93+46 = 322

85+56+86 =227

30+177+46 =253

Сума всіх доходів = 216+322+322+253 = 1018

Як бачимо, розв'язок задовольняє обмеженням та знаходить максимальн ий результат.

# ОЦІНКА ЧУТЛИВОСТІ ОТРИМАНОГО РІШЕННЯ ДО ВАРІАЦІЇ ПАРАМЕТРІВ ЗАДАЧІ

Проведемо експеримент зі варіації бюджету від 0 до 200 і за заданих даних для цього алгоритму при однакових обмеженнях $L_j \leq 30$ для всіх відділів і знайдемо результати потенціальних доходів. Після отримання масиву результатів, зобразимо графік залежності потенційного доходу від вкладеного бюджету:

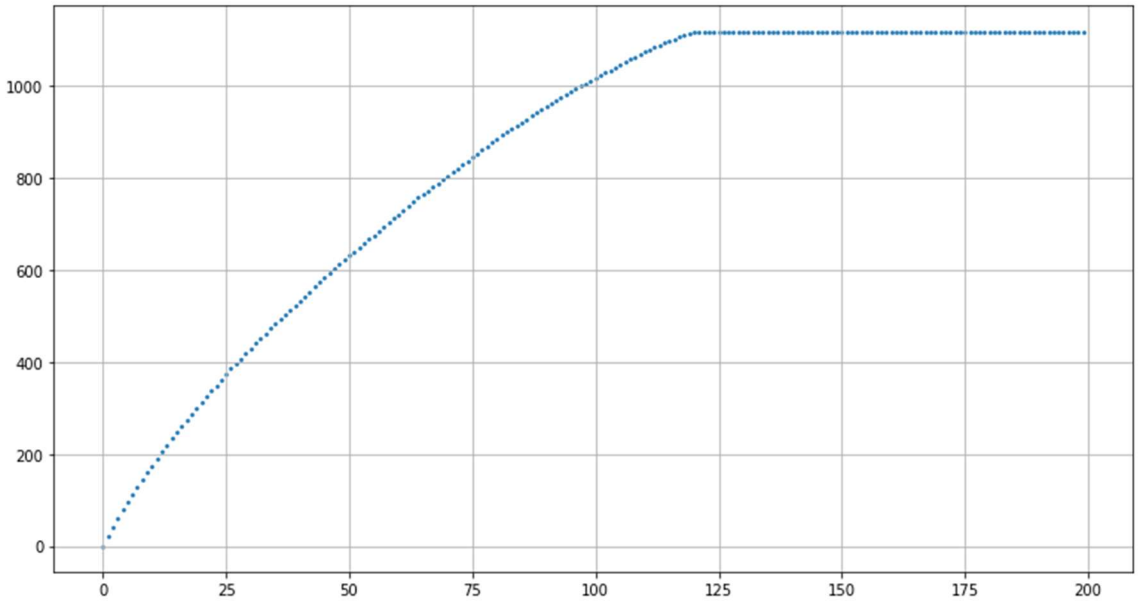


Рис. 7.1 Графік потенційного доходу від вкладеного бюджету

Як бачимо, графік цієї функції показує, що вона неспадна, зростає строго до точки 120. Це очевидно, оскільки сума обмежень на всі відділи 30+30+30+30=120. Далі за збільшення бюджету дохід рости не буде, оскільки залишаються незмінними обмеження на витрати на кожен відділ

# ВИСНОВКИ

Однією з найважливіших задач дисципліни теорії прийняття рішень є задача розподілу ресурсів за темами.. Вона є актуальною і має суттєве значення для багатьох підприємств для знаходження оптимального доходу за заданих обмежених ресурсів.

У курсовій роботі було розглянуто вирішення задачі розподілу обмежених ресурсів за темами методом динамічного програмування. У процесі роботи були побудовані відповідна математична модель задачі та запропоновано аналітичний розв’язок. На основі цього рішення було реалізовано алгоритм мультипарадигменною мовою Python3, яка надає легку підтримуваність коду для розширення функціоналу для підприємств, має гнучкий і надійний набір програмних пакетів для роботи з числами та матрицями, а також запроваджує для користувача приємний графічний користувацький інтерфейс.

У результаті роботи алгоритму було отримано необхідні аргументи, за яких буде досягнуто максимальний дохід за заданих обмежень асигнувань. Алгоритм було перевірено виконання обмежень та коректність його роботи. Крім того, було проаналізовано вплив параметрів, а саме максимальний розмір асигнувань на всі відділи на результат розв’язку задачі. Також алгоритм перевірено на адекватність роботи за часом. Складність алгоритму методом динамічного програмування набагато менше ніж варіант повного перебору, що дозволяє користувачу майже одразу побачити необхідний результат за заданих обмежень та прийняти рішення стосовно розподілу ресурсів на його підприємстві.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. Дослідження операцій, Зайченко Ю. П., 7-ме видання, 2006 р., стр. 448-468
2. Дослідження операцій. Збірник задач, Зайченко Ю. П., 8-ме видання, 2007 р., стр. 231-237

# ДОДАТОК А. ЛІСТИНГ ПРОГРАМИ

Файл solution.py

```
import pandas as pd
import numpy as np
from collections import defaultdict
from itertools import product
from tqdm import tqdm
from copy import deepcopy

class Solution:
    def __init__(self, f_list, x_list, N):
        self.f_list = f_list
        self.x_list = x_list
        self.N = N
        self._cache = defaultdict(lambda: defaultdict())
        self._argmax = defaultdict()
        self._argmax_array = []

    def lambda_(self, k, eps):
        if k in self._cache and eps in self._cache[k]:
            return deepcopy(self._cache[k][eps])
        max_result = -1
        argmax = {}
        if k==0:
            for x_k in self.x_list[k]:
                if 0<=x_k and x_k<=eps:
                    if self.f_list[k][x_k]>max_result:
                        argmax[k] = x_k
                        max_result = self.f_list[k][x_k]
        else:
            for x_k in self.x_list[k]:
                if 0<=x_k and x_k<=eps:
                    lambda_result, argmax_inner = self.lambda_(k-1, eps-
x_k)
                    result = self.f_list[k][x_k] + lambda_result
                    if result>max_result:
                        max_result = result
                        argmax_inner[k] = x_k
                        argmax = argmax_inner
```

```
        self._cache[k][eps] = max_result, argmax
        return max_result, argmax

    def solve(self):
        max_result, argmax = self.lambda_(len(self.f_list)-1, self.N)
        return max_result, argmax


class ThemeSolver:
    def __init__(self, phi_list, theta_list, L_list, N):
        self.phi_list = phi_list
        self.theta_list = theta_list
        self.L_list = L_list
        self.N = N
        self._cache = defaultdict(lambda: defaultdict())
        self._argmax = defaultdict()

    def dep_lambda_(self, k, eps):
        if k in self._cache and eps in self._cache[k]:
            return deepcopy(self._cache[k][eps])
        max_result = -1
        argmax = {}
        if k==0:
            solution = Solution(self.phi_list[k], self.theta_list[k],
min(eps, self.L_list[k]))
            phi_result, phi_argmax = solution.solve()
            argmax[k] = phi_argmax
            max_result = phi_result
        else:
            for gamma_k in range(min(self.L_list[k]+1,eps+1)):
                lambda_result, inner_argmax = self.dep_lambda_(k-1, eps-
gamma_k)
                sol = Solution(self.phi_list[k], self.theta_list[k],
gamma_k)
                phi_result, phi_argmax = sol.solve()
                result = phi_result + lambda_result
                if result>max_result:
                    max_result=result
                    inner_argmax[k] = phi_argmax
                    argmax = inner_argmax
```

```
        self._cache[k][eps] = max_result, argmax
        return max_result, argmax

    def solve(self):
        max_result, argmax = self.dep_lambda_(len(self.phi_list)-1,
self.N)
        return max_result, argmax
```

Файл utils.py

```
import numpy as np
import pandas as pd

def parse_data(path):
    department_1 = pd.read_excel("./data/data.xlsx",sheet_name=0)
    department_2 = pd.read_excel("./data/data.xlsx",sheet_name=1)
    department_3 = pd.read_excel("./data/data.xlsx",sheet_name=2)
    department_4 = pd.read_excel("./data/data.xlsx",sheet_name=3)

    department_1_p_v =
drop_missing(department_1,"v","P_1_v").set_index("v")["P_1_v"].to_dict()
    department_1_q_w =
drop_missing(department_1,"w","Q_1_w").set_index("w")["Q_1_w"].to_dict()
    department_1_r_x =
drop_missing(department_1,"x","R_1_x").set_index("x")["R_1_x"].to_dict()

    department_2_p_v =
drop_missing(department_2,"v","P_2_v").set_index("v")["P_2_v"].to_dict()
    department_2_q_w =
drop_missing(department_2,"w","Q_2_w").set_index("w")["Q_2_w"].to_dict()
    department_2_r_x =
drop_missing(department_2,"x","R_2_x").set_index("x")["R_2_x"].to_dict()

    department_3_p_v =
drop_missing(department_3,"v","P_3_v").set_index("v")["P_3_v"].to_dict()
    department_3_q_w =
drop_missing(department_3,"w","Q_3_w").set_index("w")["Q_3_w"].to_dict()
    department_3_r_x =
drop_missing(department_3,"x","R_3_x").set_index("x")["R_3_x"].to_dict()

    department_4_p_v =
drop_missing(department_4,"v","P_4_v").set_index("v")["P_4_v"].to_dict()
    department_4_q_w =
drop_missing(department_4,"w","Q_4_w").set_index("w")["Q_4_w"].to_dict()
    department_4_r_x =
drop_missing(department_4,"x","R_4_x").set_index("x")["R_4_x"].to_dict()

    department_1_p_v_investments =
np.array(sorted(department_1_p_v.keys()))
```

```
    department_1_q_w_investments =
np.array(sorted(department_1_q_w.keys()))
    department_1_r_x_investments =
np.array(sorted(department_1_r_x.keys()))

    department_2_p_v_investments =
np.array(sorted(department_2_p_v.keys()))
    department_2_q_w_investments =
np.array(sorted(department_2_q_w.keys()))
    department_2_r_x_investments =
np.array(sorted(department_2_r_x.keys()))

    department_3_p_v_investments =
np.array(sorted(department_3_p_v.keys()))
    department_3_q_w_investments =
np.array(sorted(department_3_q_w.keys()))
    department_3_r_x_investments =
np.array(sorted(department_3_r_x.keys()))

    department_4_p_v_investments =
np.array(sorted(department_4_p_v.keys()))
    department_4_q_w_investments =
np.array(sorted(department_4_q_w.keys()))
    department_4_r_x_investments =
np.array(sorted(department_4_r_x.keys()))

    department_1_investments = [department_1_p_v_investments,
                                department_1_q_w_investments,
                                department_1_r_x_investments]

    department_2_investments = [department_2_p_v_investments,
                                department_2_q_w_investments,
                                department_2_r_x_investments]

    department_3_investments = [department_3_p_v_investments,
                                department_3_q_w_investments,
                                department_3_r_x_investments]

    department_4_investments = [department_4_p_v_investments,
                                department_4_q_w_investments,
                                department_4_r_x_investments]
```

```
    department_1_profits = [department_1_p_v,
                            department_1_q_w,
                            department_1_r_x]
    department_2_profits = [department_2_p_v,
                            department_2_q_w,
                            department_2_r_x]
    department_3_profits = [department_3_p_v,
                            department_3_q_w,
                            department_3_r_x]
    department_4_profits = [department_4_p_v,
                            department_4_q_w,
                            department_4_r_x]

    profits_list = [department_1_profits,
                    department_2_profits,
                    department_3_profits,
                    department_4_profits]

    investments_list = [department_1_investments,
                        department_2_investments,
                        department_3_investments,
                        department_4_investments]

    return profits_list, investments_list

def drop_missing(department_data:pd.DataFrame, key_colname:str,
value_colname:str, missing="___"):
    return
department_data[department_data[value_colname]!="___"][[key_colname,value
_colname]]

def write_report_to_file(report:str, path:str):
    with open(path,"w") as f:
        f.write(report)

def get_report(profits_list, investments_list,
department_max_investments, budget, solution):
    report = ""
    report+=f"Загальна сума асигнувань на всі проекти = {budget}"+"\n"
```

```
    max_investments_string = " ".join([f"L_{i+1}={max_investment}" for
i,max_investment in enumerate(department_max_investments)])
    report+=f"Верхні межі асигнувань відділів:
"+max_investments_string+"\n"
    result, argmax = solution
    report+=f"Максимальний результат = {result}"+"\n"
    indexes_strings = []
    functions_strings = []
    cases_names = ["P","Q","R"]
    cases_args_numbers = ["V","W","X"]
    overall_score = 0
    overall_investment = 0
    for department_id in argmax:
        indexes_string = ""
        functions_string = ""
        department_profit = 0
        department_investment = 0
        for case_id in solution[1][department_id]:
            investment = solution[1][department_id][case_id]
            profit = profits_list[department_id][case_id][investment]
            department_profit += profit
            department_investment += investment
            functions_string+="{}({}) =
{}    ".format(cases_names[case_id], investment, profit)
            indexes_string+=f"{cases_args_numbers[case_id]}={investment}
"
        functions_string += "| Sum={}".format(department_profit)
        indexes_string += "| Sum={}".format(department_investment)
        indexes_strings.append(indexes_string)
        functions_strings.append(functions_string)
        overall_score+=department_profit
        overall_investment += department_investment
    report += "\n"
    report += "\n".join(indexes_strings)+"\n"
    report += f"Сума всіх інвестицій = {overall_investment}"+"\n\n"
    report += "\n".join(functions_strings)+"\n"
    report += f"Сума всіх доходів = {overall_score}"
    return report
```

Файл coursework.ipynb (графічний інтерфейс)

In [1]:
```
import numpy as np
import pandas as pd
from collections import defaultdict
```

In [2]:
```
from solution import ThemeSolver
from utils import get_report, drop_missing, write_report_to_file, parse_data
```

In [3]:
```
%load_ext autoreload
%autoreload 2
```

In [4]:
```
profits_list, investments_list = parse_data("data/data.xlsx")
```

In [6]:
```
L1 = int(input("Введіть L1: "))
L2 = int(input("Введіть L2: "))
L3 = int(input("Введіть L3: "))
L4 = int(input("Введіть L4: "))

N = int(input("Введіть N: "))
L = [L1,L2,L3,L4]
```

```
Введіть L1: 30
Введіть L2: 30
Введіть L3: 30
Введіть L4: 30
Введіть N: 100
```

In [7]:
```
solver = ThemeSolver(profits_list, investments_list, L, N)
```

In [8]:
```
%%time
solution = solver.solve()
```

```
Wall time: 7.8 s
```

In [9]:
```
report = get_report(profits_list, profits_list, L, N, solution)
```

In [10]:
```
print(report)
```

```
Загальна сума асигнувань на всі проекти = 100
Верхні межі асигнувань відділів: L_1=30 L_2=30 L_3=30 L_4=30
Максимальний результат = 1018

V=5 W=8 X=11 | Sum=24
V=15 W=9 X=5 | Sum=29
V=9 W=7 X=8 | Sum=24
V=4 W=14 X=5 | Sum=23
Сума всіх інвестицій = 100

P(5) = 36    Q(8) = 67    R(11) = 113    | Sum=216
P(15) = 183    Q(9) = 93    R(5) = 46    | Sum=322
P(9) = 85    Q(7) = 56    R(8) = 86    | Sum=227
P(4) = 30    Q(14) = 177    R(5) = 46    | Sum=253
Сума всіх доходів = 1018
```

In [11]:
```
write_report_to_file(report,"report.txt")
```