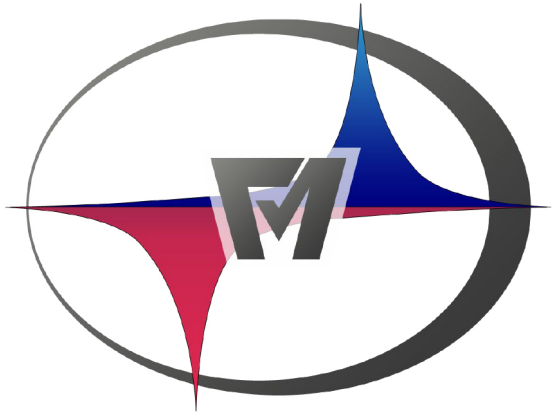


Парадигма процедурного програмування

к.т.н., доцент кафедри прикладної математики
Рижа Ірина Андріївна

Про що ця лекція???

- Опишемо модульну структуру алгоритму.
- Викладемо особливості процедурного програмування.

Модульна структура алгоритму

Модульний принцип побудови алгоритмів

полягає в розбитті початкової задачі на більш прості підзадачі (підалгоритми).

- Алгоритми для підзадач будуються достатньо просто;
- Уже існують відповідні побудовані та протестовані алгоритми;
- Розрахунок на майбутнє використання як складових частин більш складних задач.

Модуль

– алгоритм, який розрахований на багаторазове використання, як складова частина при побудові алгоритмів різних задач, і є оформлений відповідним чином.

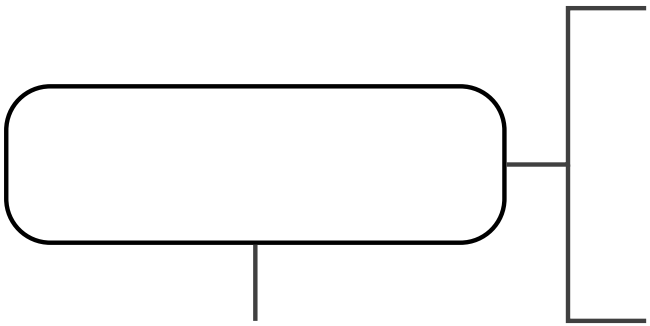
Вимоги до модулів

1. Простота

Кожен модуль – це алгоритм розв'язання окремої задачі.

2. Список формальних параметрів

Модуль повинен мати один вхід (вхідні дані для роботи модуля) і один вихід (результати роботи модуля). Порядок передачі модулю вхідних даних і повернення результатів роботи є строго обумовленим.



Ім'я модуля та
список **формальних** параметрів

Вимоги до модулів

3. Самодостатність

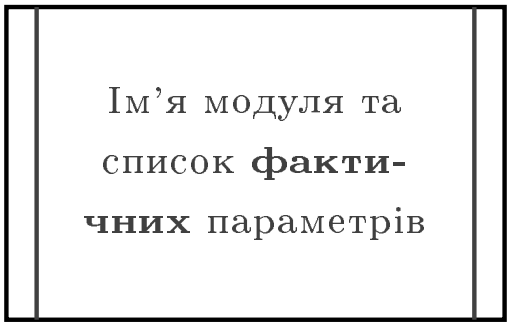
Модуль не залежить від інших модулів і його активізація проводиться шляхом виклику і передачі **фактичних** параметрів

4. Принцип “екранування” змінних

Змінні величини, які використовуються модулем, як правило, це не однойменні змінні інших модулів.

Звертання до модулів

Символ **“попередньо створених і окремо описаних алгоритмів і програм”**:

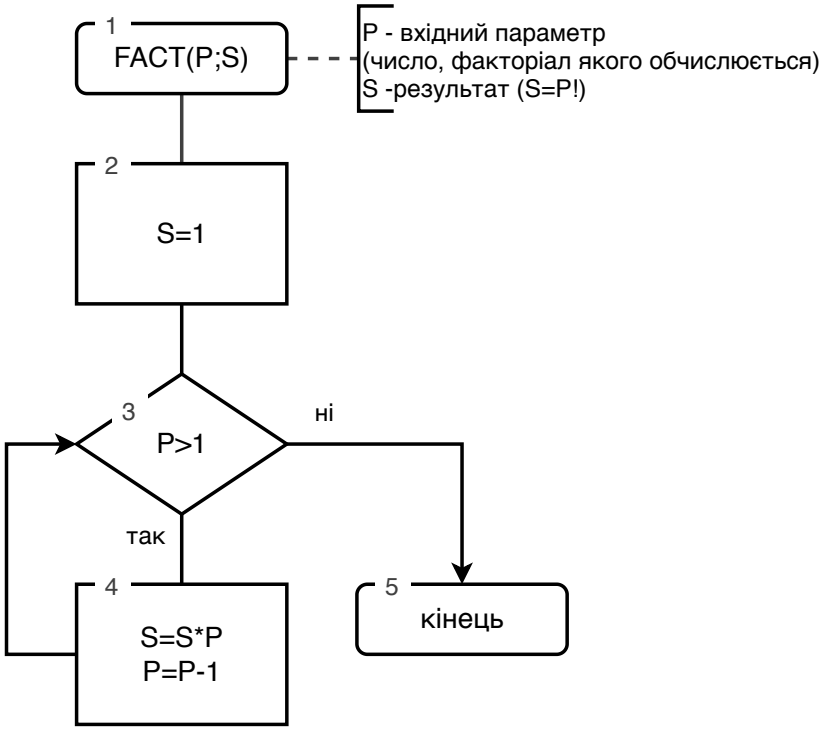


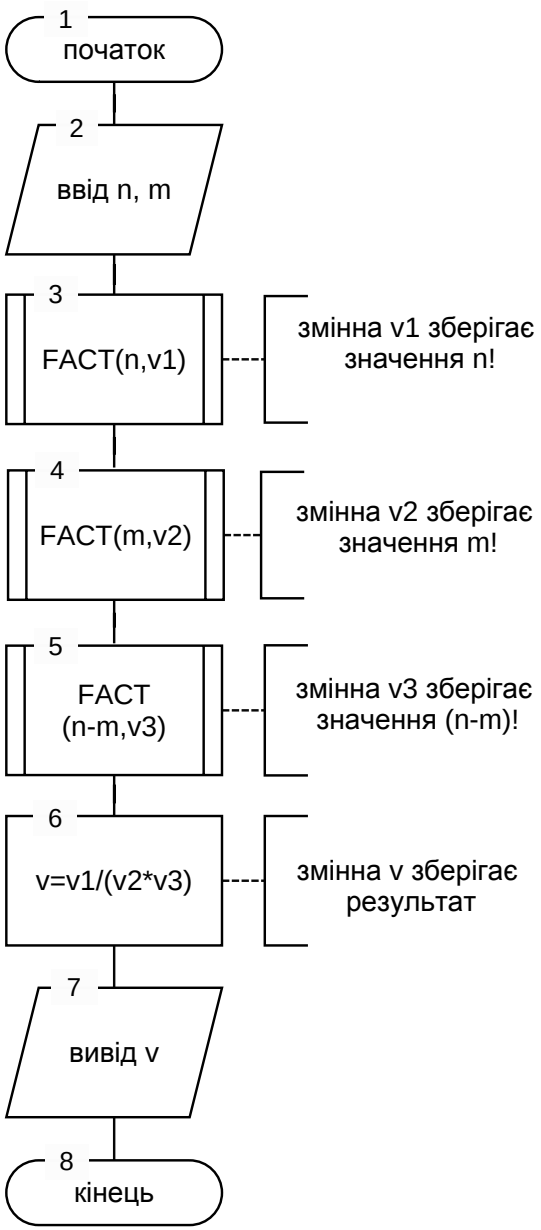
Приклад 1.

Побудувати алгоритм обчислення числа сполук з n елементів множини по m:

$$\binom{n}{m} = \frac{n!}{m!(n-m)!}.$$

- $FACT\,(P, S)$ – модуль обчислення $n!$;
- P – вхідний параметр – число, факторіал якого обчислюється;
- S – вихідний параметр – результат $(S = P!)$


1
FACT(P;S)
P - вхідний параметр
(число, факторіал якого обчислюється)
S -результат (S=P!)
2
S=1
3
P>1
ні
так
4
S=S*P
P=P-1
5
кінець


1
початок
2
ввід n, m
3
FACT(n,v1)
змінна v1 зберігає значення n!
4
FACT(m,v2)
змінна v2 зберігає значення m!
5
FACT (n-m,v3)
змінна v3 зберігає значення (n-m)!
6
v=v1/(v2*v3)
змінна v зберігає результат
7
вивід v
8
кінець

Поміркуймо...

Чи буде описаний вище алгоритм ефективним???

Принцип процедурного програмування

Процедурне (структурне) програмування

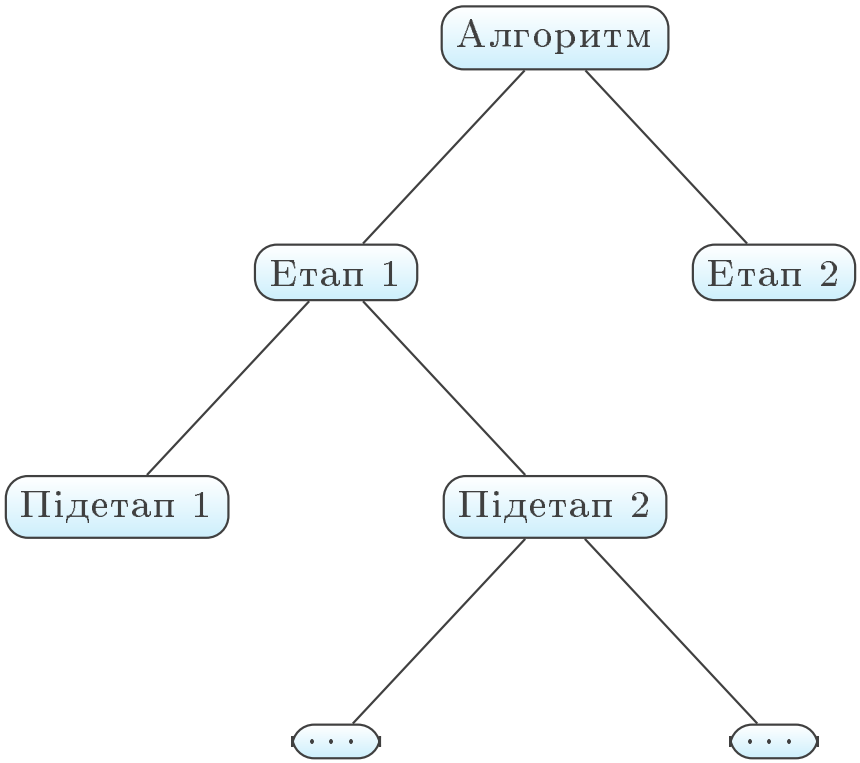
– це поетапна розробка та уточнення алгоритму.

Метод проектування “зверху-вниз” або метод покрокової деталізації

– метод проектування та розробки алгоритму згідно якого

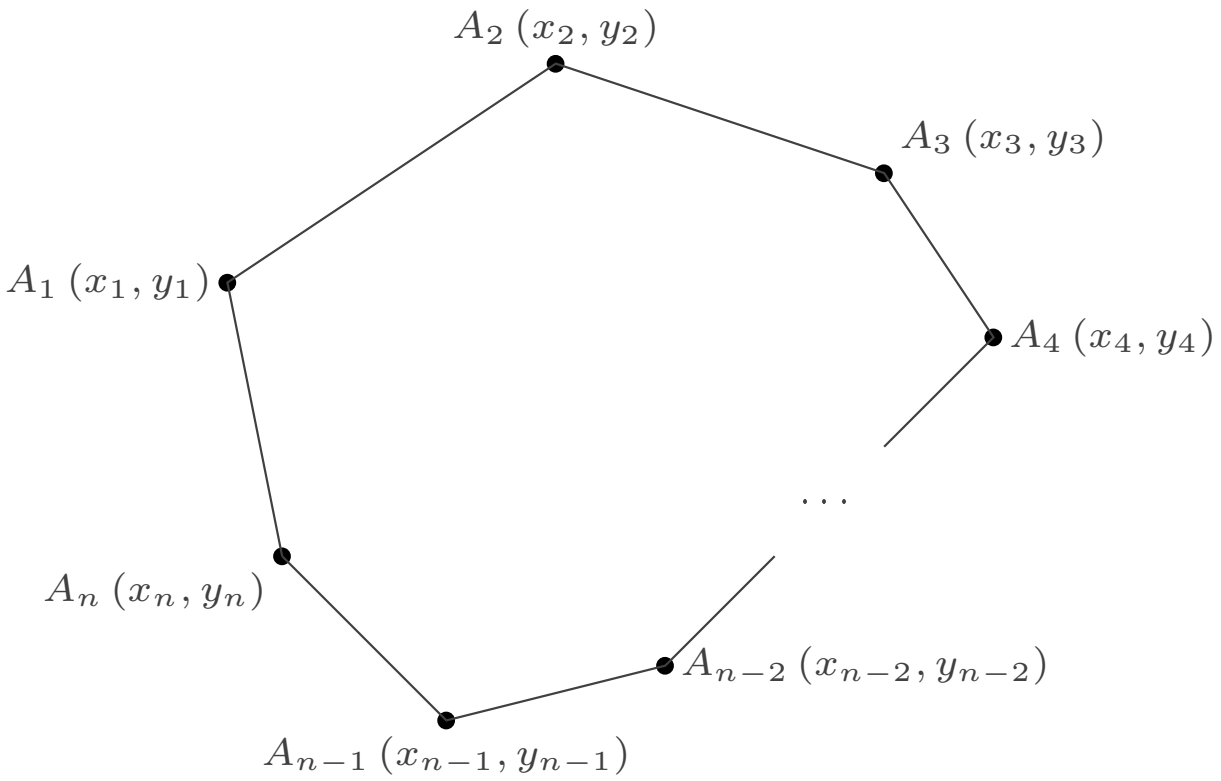
1. будується *узагальнена структура алгоритму* – алгоритм розбивається на декілька етапів (підзадач);
2. кожен з етапів деталізується і оформлюється у вигляді модуля;
   - етапи в свою чергу можуть розбиватись на підетапи і уточнюватись;
3. відбувається процес “згортання” етапів та підетапів у єдиний алгоритм.

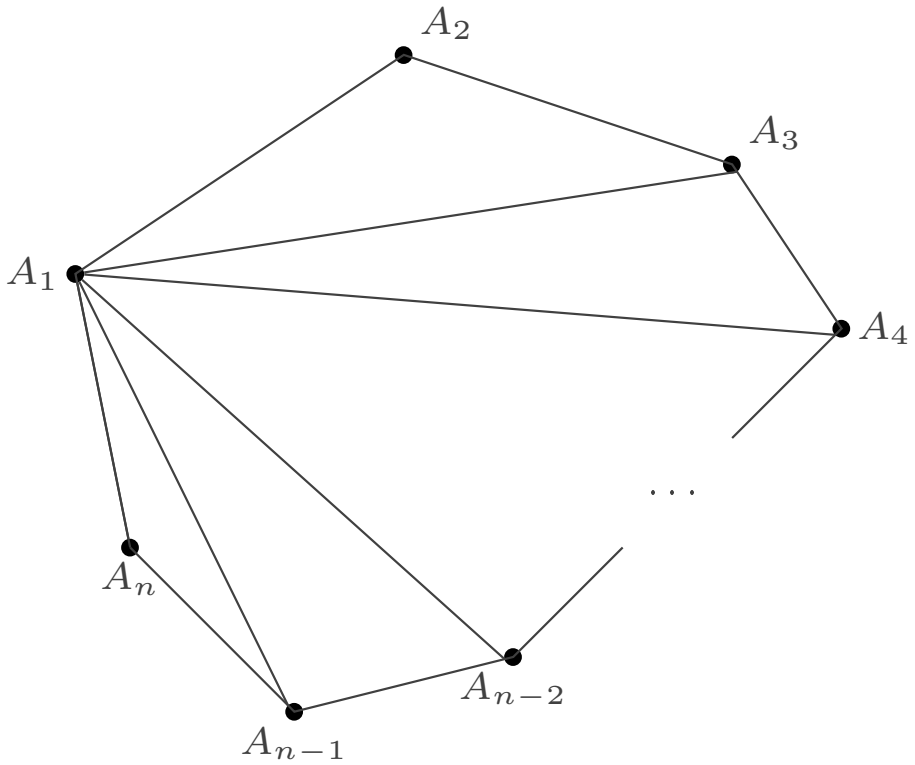
Схема методу проектування “зверху-вниз”



Приклад 2.

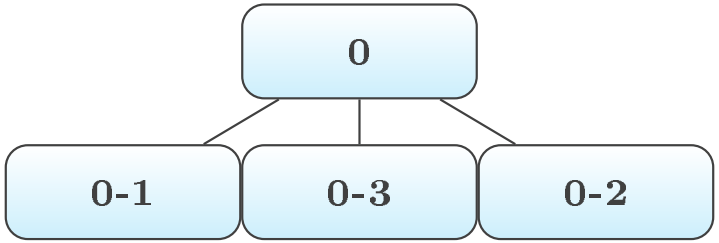
Побудувати алгоритм обчислення площі випуклого n-кутника, заданого координатами своїх вершин на площині.





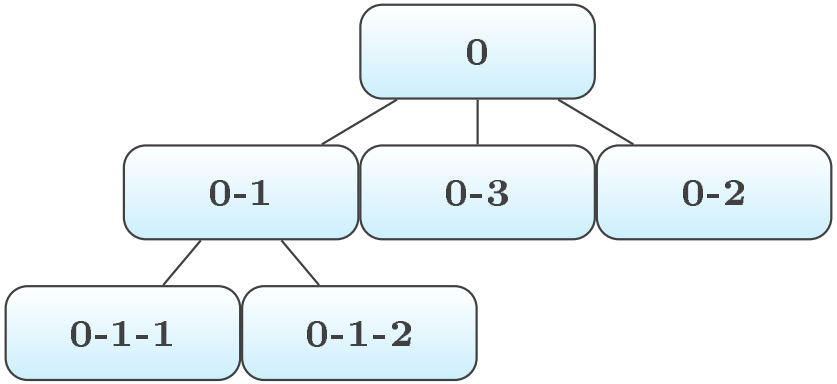
Площу багатокутника $A_1A_2A_3\ldots A_n$ можна знайти як суму площ трикутників

$$A_1A_iA_{i+1}, \quad i = \overline{2, n-1}.$$



0: обчислити площу випуклого n-кутника, заданого координатами своїх вершин:

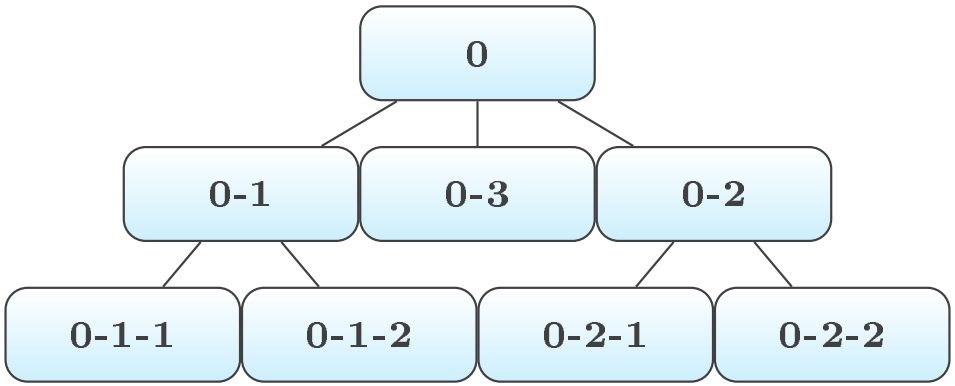
- **0-1:** ввести координати вершин n-кутника;
- **0-2:** обчислити площу n-кутника;
- **0-3:** вивести результати обчислень.



0-1: ввести координати вершин n-кутника:

0-1-1: задати число вершин багатокутника – ціле число n;

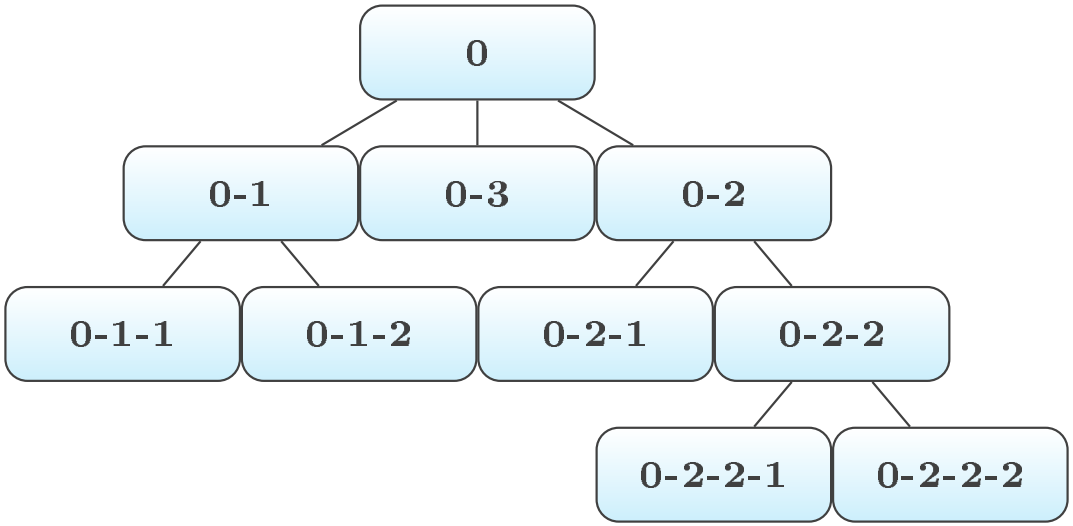
0-1-2: ввести координати вершин n-кутника у масив $A(n, 2)$;



0-2: обчислити площу n-кутника:

0-2-1: розбити n-кутник на трикутники $A_1A_iA_{i+1}$, $i = \overline{2, n-1}$;

0-2-2: обчислити площу трикутника $A_1A_iA_{i+1}$;



0-2-2: обчислити площу трикутника $A_1 A_i A_{i+1}$:

0-2-2-1: обчислити за заданими номерами вершин довжину сторони трикутника:

$$A_i\left(x_i; y_i\right), A_{i+1}\left(x_{i+1}; y_{i+1}\right) : l_{A_i A_{i+1}} = \sqrt{\left(x_{i+1} - x_i\right)^2 + \left(y_{i+1} - y_i\right)^2};$$

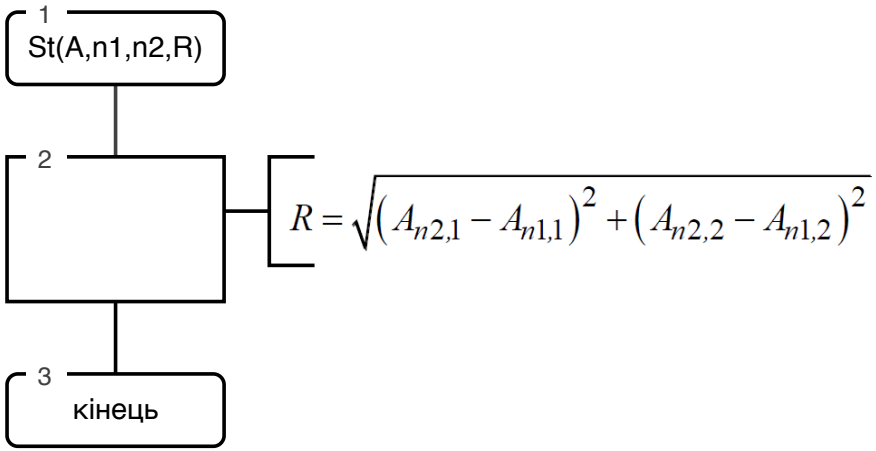
0-2-2-2: обчислити площу трикутника за формулою Герона:

$$S = \sqrt{p\left(p-a\right)\left(p-b\right)\left(p-c\right)}, \quad p = \frac{\left(a+b+c\right)}{2}.$$

Модуль обчислення довжини відрізка (0-2-2-1)

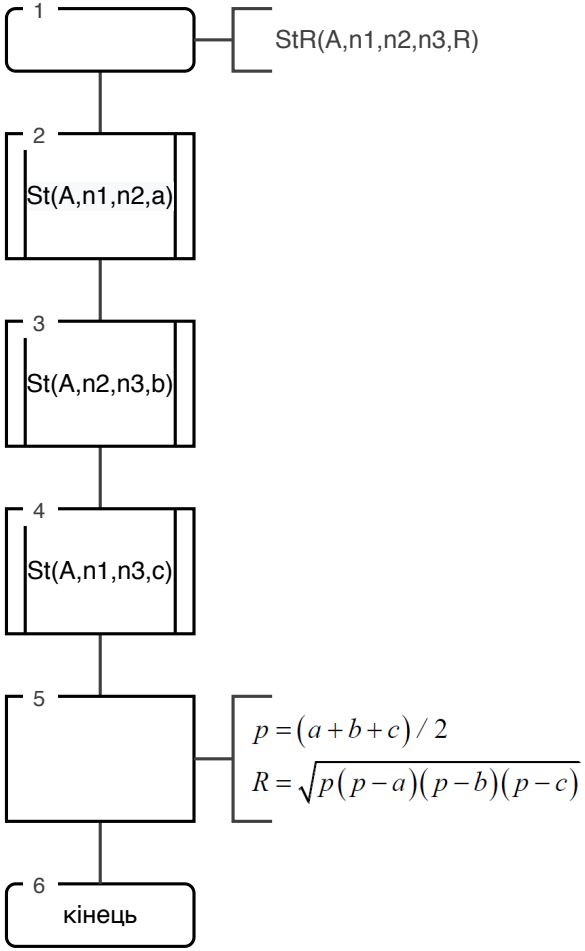
- A $(n\times 2)$ – масив координат вершин n-кутника;
- $n1$, $n2$ – номери вершин;
- R – довжина відрізка $A_{n1}A_{n2}$.

	1	2
1	x_1	y_1
2	x_2	y_2
$\vdots$	$\vdots$	$\vdots$
n	x_n	y_n



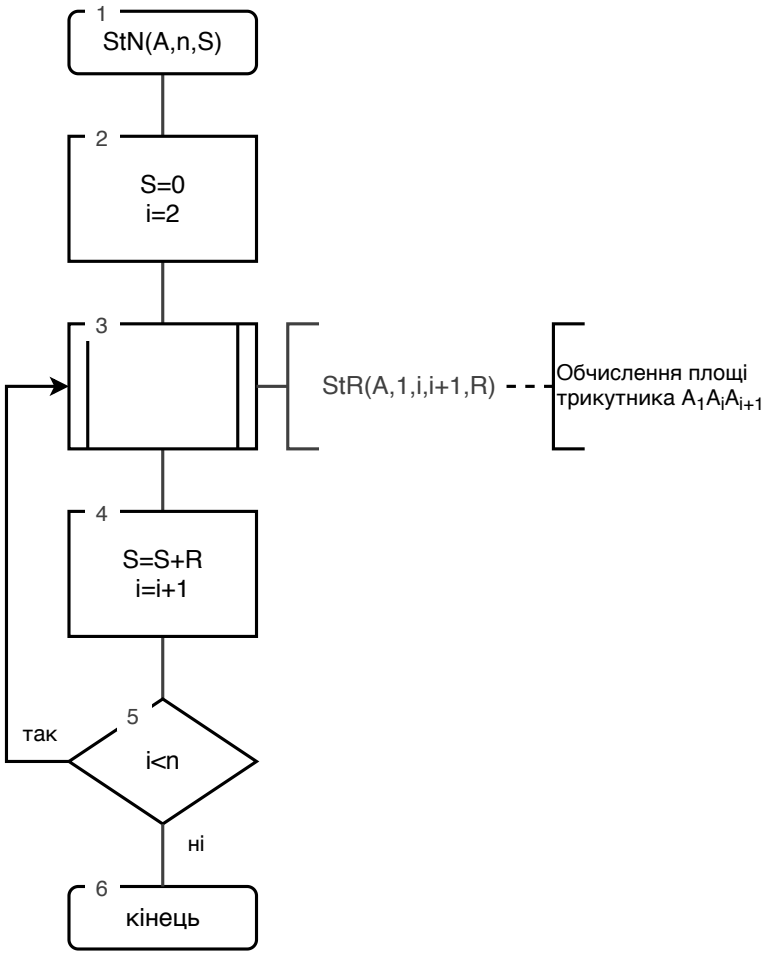
Модуль обчислення площі трикутника (0-2-2)

- A $(n\times 2)$ – масив координат вершин n-кутника;
- $n1$, $n2$, $n3$ – номери вершин;
- R – площа трикутника $A_{n1}A_{n2}A_{n3}$.

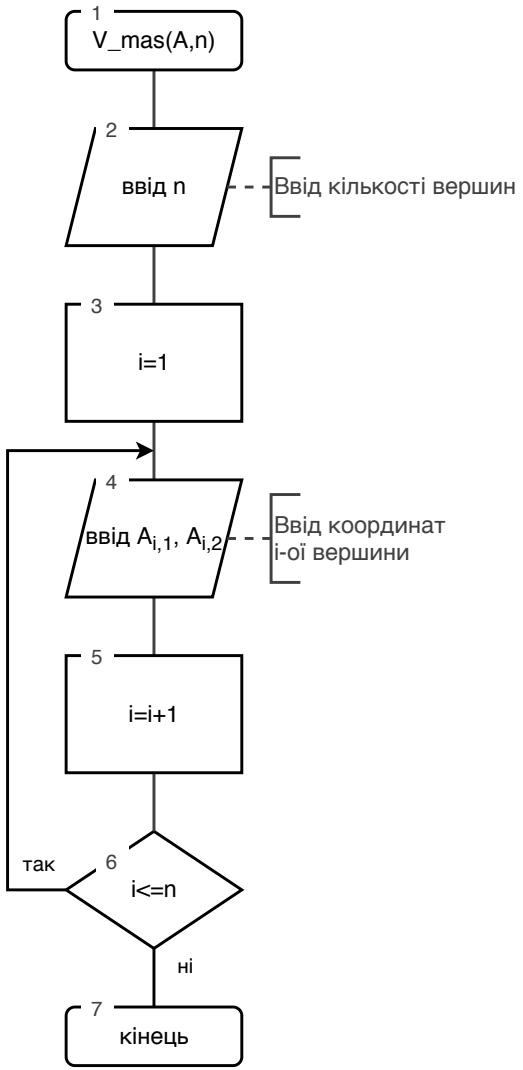


Модуль обчислення площі n-кутника (0-2)

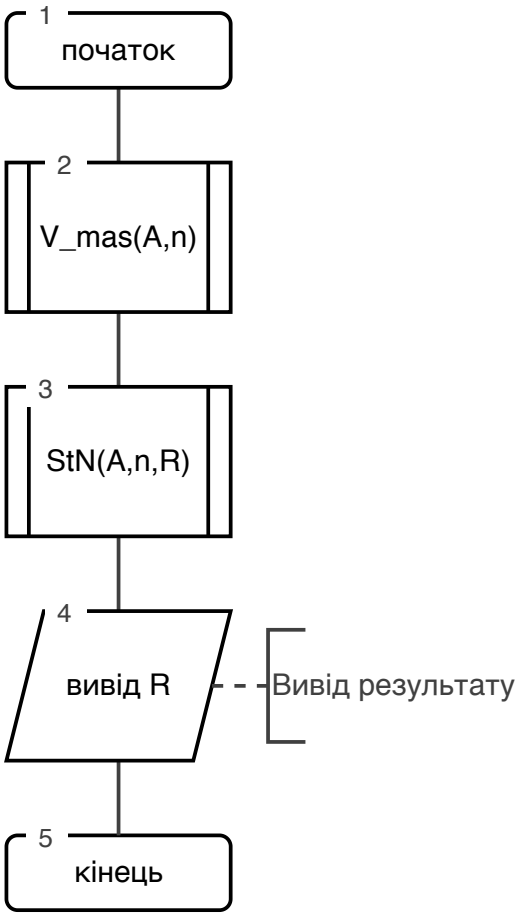
- A $(n \times 2)$ – масив координат вершин n-кутника;
- S – площа n-кутника.



Модуль вводу координат вершин (0-1)



Загальний алгоритм



Приклад 3.

Обчислити значення матричного многочлена:

$$\sum_{i=0}^{k} b_i A^{k-i} = b_0 A^k + b_1 A^{k-1} + \cdots + b_{k-1} A + b_k I,$$

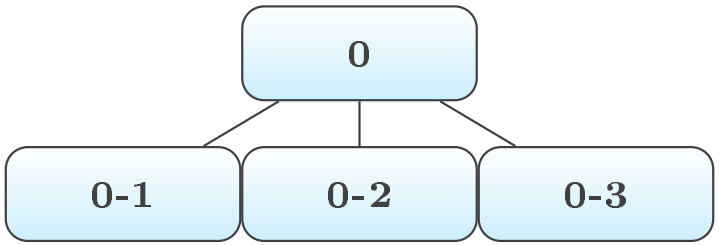
де A – квадратна матриця розмірності $n \times n$;

b_i, $i = \overline{0, k}$ – коефіцієнти;

I – одинична матриця розмірності $n \times n$.

$$\sum_{i=0}^{k} b_i A^{k-i} = b_0 A^k + b_1 A^{k-1} + \cdots + b_{k-1} A + b_k I =$$

$$= \Big(\ldots \Big(\Big(\underbrace{b_0 I}_{\text{ядро}} A + b_1 I \Big) A + b_2 I \Big) A + \ldots + b_{k-1} I \Big) A + b_k I$$

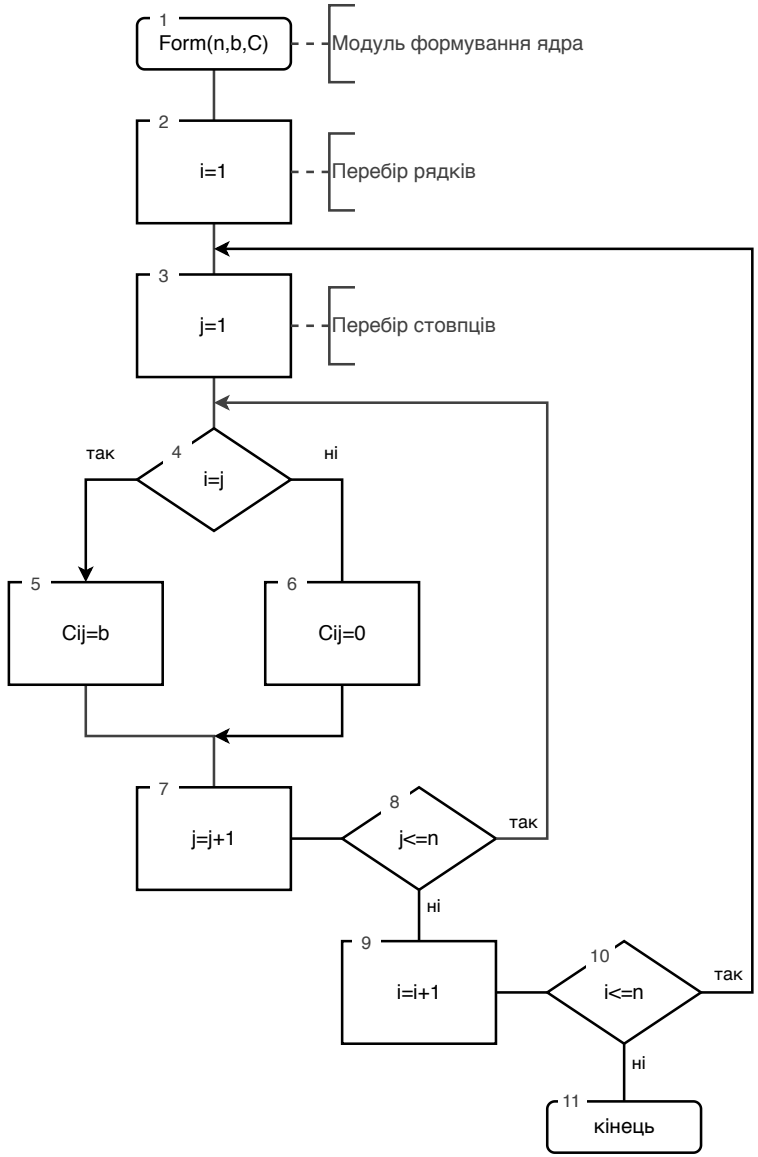


0: обчислити значення матричного многочлена:

- **0-1:** сформувати **ядро** $C = b_0 I$;
- **0-2:** перемножити матриці $D = C \cdot A$;
- **0-3:** додати матриці $D + b_i I$.

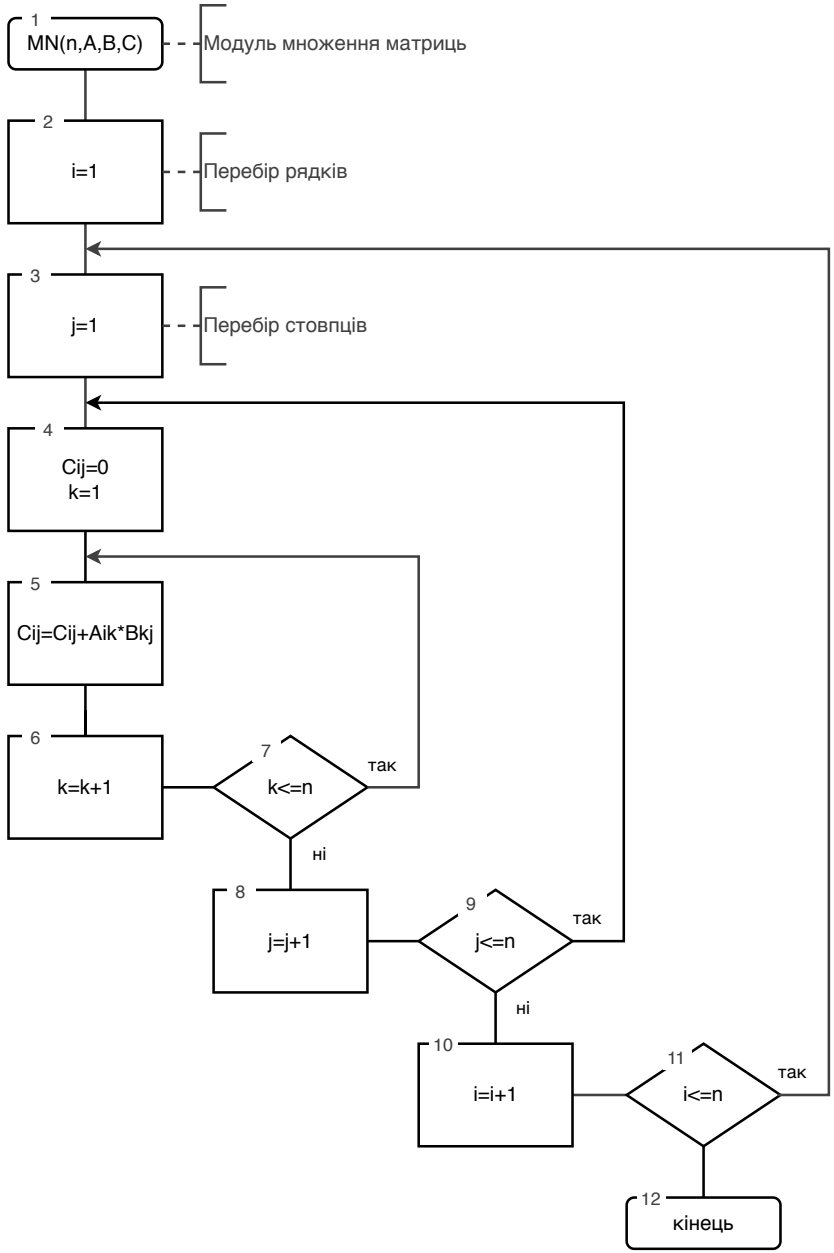
- n – розмірність матриці;
- b – коефіцієнт при найбільшому степені;
- C $(n \times n)$ – матриця-ядро.

$$C = b_0 I = \begin{pmatrix} b_0 & 0 & \dots & 0 \\ 0 & b_0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & b_0 \end{pmatrix}$$



- n – розмірність;
- A, B – дві матриці розмірності $n \times n$, які потрібно перемножити;
- C $(n \times n)$ – матриця-результат.

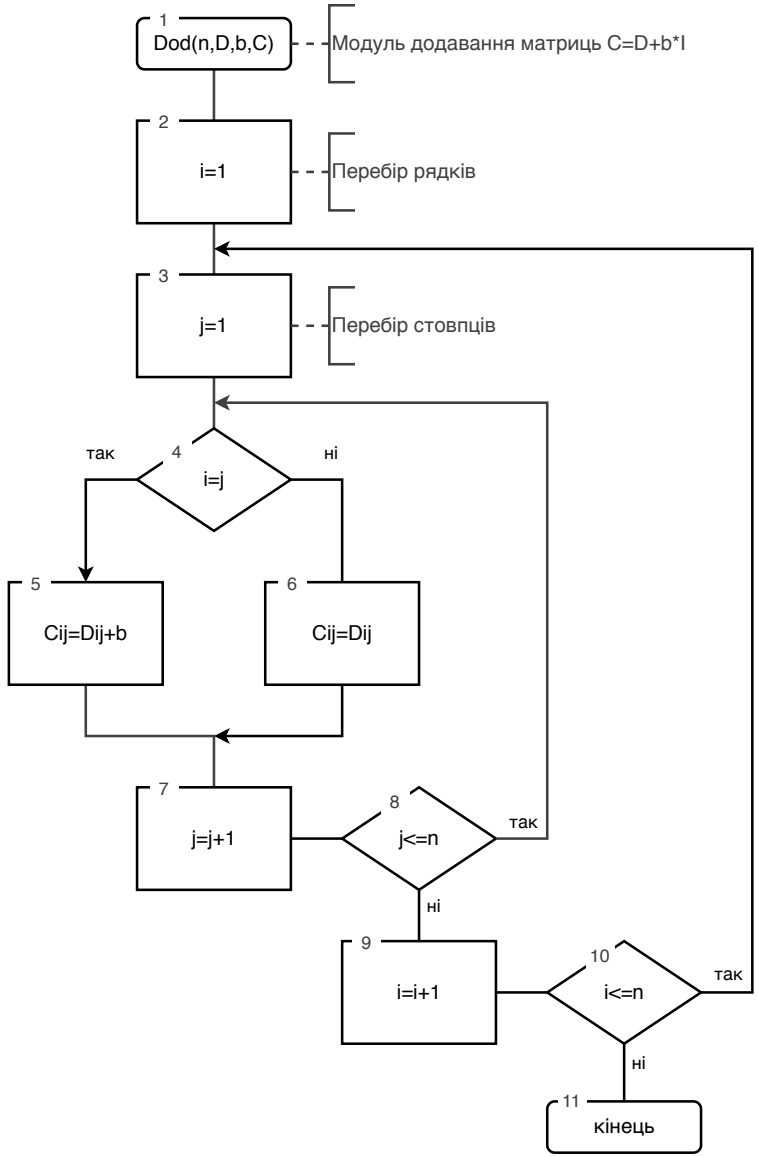
$$c_{ij} = \sum_{k=1}^{n} a_{ik} \cdot b_{kj}$$



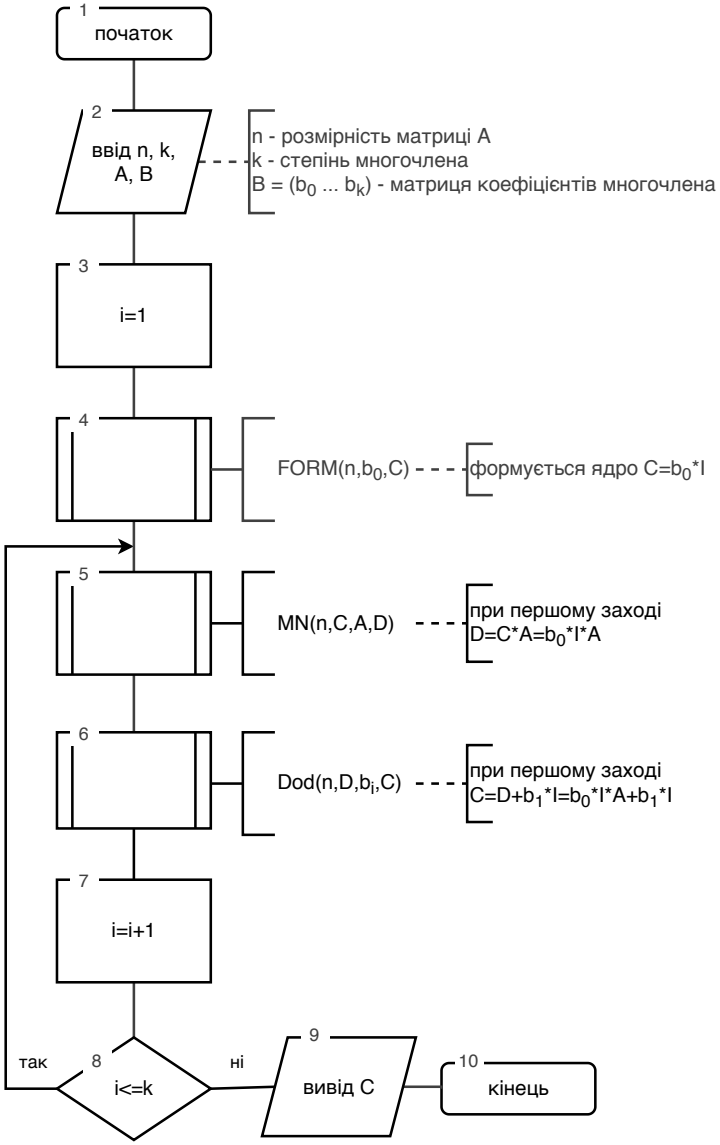
- n – розмірність;
- D – матриця розмірності $n \times n$;
- b – коефіцієнт (число);
- C – матриця-результат.

$$C = D + b_i \cdot I =$$

$$= \begin{pmatrix} d_{11} + b_i & d_{12} & \dots & d_{1n} \\ d_{21} & d_{11} + b_i & \dots & d_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{n1} & d_{n2} & \dots & d_{nn} + b_i \end{pmatrix}$$



Основна програма



Дякую за увагу!

Далі буде...