

Міністерство освіти і науки України  
Національний технічний університет України «Київський політехнічний  
інститут імені Ігоря Сікорського»  
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт  
з лабораторної роботи № 6 з  
дисципліни «Алгоритми та структури  
даних-1. Основи алгоритмізації»  
«Дослідження рекурсивних  
алгоритмів»  
Варіант 2

Виконав студент ПІ-12, Басараб Олег Андрійович  
(шифр, прізвище, ім'я, по батькові)

Перевірів \_\_\_\_\_  
(прізвище, ім'я, по батькові)

## Лабораторна робота №6 “Дослідження рекурсивних алгоритмів”

### Варіант 2

**Мета** – дослідити особливості роботи рекурсивних алгоритмів та набути практичних навичок їх використання під час складання програмних специфікацій підпрограм.

**Задача 2.** Обчислення добутку елементів геометричної прогресії, що убуває:

початкове значення – 64, кінцеве значення – 1, крок –  $\frac{1}{4}$ .

### Розв'язок

**Постановка задачі.** Результатом розв'язку є дійсна змінна `termsProduct`. Для його знаходження вхідні дані не потрібні. Початковими даними є дійсні константи `FIRST_TERM = 64`, `LAST_TERM = 1` та `COMMON_RATIO = 0.25`.

**Побудова математичної моделі.** Складемо таблицю імен змінних основної функції.

Змінна	Тип	Ім'я	Призначення
Перший елемент геометричної прогресії	Дійсний (константа)	FIRST_TERM	Початкове дане
Останній елемент геометричної прогресії	Дійсний (константа)	LAST_TERM	Початкове дане
Знаменник геометричної прогресії	Дійсний (константа)	COMMON_RATIO	Початкове дане
Добуток елементів геометричної прогресії з першого по останній	Дійсний	termsProduct	Результат

Складемо таблицю імен змінних функції findNthTerm для знаходження N-ого елемента геометричної прогресії.

Змінна			Тип	Ім'я	Призначення
Індекс	шуканого	елементу	Цілий	n	Вхідне дане
геометричної прогресії					
Останній	елемент	геометричної	Дійсний	firstTerm	Вхідне дане
прогресії					
Знаменник геометричної прогресії			Дійсний	commonRatio	Вхідне дане
Шуканий	елемент	геометричної	Дійсний	nthTerm	Результат
прогресії					

Складемо таблицю імен змінних функції findTermsProduct для знаходження добутку елементів геометричної прогресії з першого по останній.

Змінна			Тип	Ім'я	Призначення
Перший	елемент	геометричної	Дійсний	firstTerm	Вхідне дане
прогресії					
Останній	елемент	геометричної	Дійсний	lastTerm	Вхідне дане
прогресії					
Знаменник геометричної прогресії			Дійсний	commonRatio	Вхідне дане
Індекс	поточного	елементу	Дійсний	termIndex	Поточне дане
геометричної прогресії					
Поточний	елемент	геометричної	Дійсний	currentTerm	Поточне дане
прогресії					
Добуток	елементів	геометричної	Дійсний	termsProduct	Результат
прогресії з першого по останній					

Таким чином, формулювання завдання зводиться до:

- 1. Опису функції `findNthTerm` для знаходження N-ого елементу геометричної прогресії.** Вхідними даними функції є ціле число `n` та дійсні числа `firstTerm`, `commonRatio`. Оскільки алгоритм роботи функції є рекурсивним, то в тілі функції скористаємось альтернативною формою оператора вибору з умовою (`n != 1`). Якщо умова є справедливою, то виконується рекурсивна гілка з виразом `nthTerm = findNthTerm(n - 1, firstTerm, commonRatio) * commonRatio`, а якщо ні, то термінальна гілка з виразом `nthTerm = firstTerm`. Такий алгоритм спирається на формулу для знаходження N-ого елементу геометричної прогресії  $b_n = b_{n-1} \cdot q$ . Результатом виконання функції є змінна `nthTerm`.
- 2. Опису функції `findTermsProduct` для знаходження добутку елементів геометричної прогресії з першого по останній.** Вхідними даними функції є дійсні числа `firstTerm`, `lastTerm`, `commonRatio`. В тілі функції відбувається ініціалізація змінних `termIndex = 1` (бо обчислення добутку елементів починаємо з першого) та `termsProduct = 1` (бо для знаходження `termsProduct` використовуватимемо рекурсивний вираз вигляду `termsProduct = termsProduct * x`). Далі знаходимо `termsProduct` в ітераційному циклі з постумовою з умовою невиходу `currentTerm != lastTerm`. В тілі циклу знаходимо поточний елемент геометричної прогресії (`currentTerm = findNthTerm(termIndex, firstTerm, commonRatio)`), домножуємо `termsProduct` на `currentTerm` (`termsProduct *= currentTerm`) та збільшуємо індекс поточного елементу геометричної прогресії на 1 (`termIndex += 1`). Результатом виконання функції є змінна `termsProduct`.
- 3. Опису основної функції.** В ній знаходимо значення змінної `termsProduct` за допомогою функції `findTermsProduct` (`termsProduct = findTermsProduct(FIRST_TERM, LAST_TERM, COMMON_RATIO)`) та виводимо його.

Розіб'ємо алгоритм роботи основної функції на кроки:

*Крок 1.* Визначимо основні дії.

*Крок 2.* Деталізуємо дію знаходження добутку елементів геометричної прогресії з першого по останній за допомогою функції `findTermsProduct`.

Розіб'ємо алгоритм роботи функції `findNthTerm` для знаходження N-ого елементу геометричної прогресії:

*Крок 1.* Визначимо основні дії.

*Крок 2.* Деталізуємо дію знаходження N-ого елементу геометричної прогресії за допомогою рекурсії.

Розіб'ємо алгоритм роботи функції `findTermsProduct` для знаходження добутку елементів геометричної прогресії з першого по останній:

*Крок 1.* Визначимо основні дії.

*Крок 2.* Деталізуємо дію ініціалізації змінних, що використовуватимуться при роботі функції.

*Крок 2.* Деталізуємо дію знаходження добутку елементів геометричної прогресії з першого по останній за допомогою ітераційного циклу з постумовою.

Програмні специфікації запишемо у псевдокоді та в графічній формі в блок-схемі алгоритму.

## Псевдокод.

### Основна функція:

*Крок 1*

**початок**

знаходження добутку елементів  
геометричної прогресії з першого  
по останній за допомогою  
функції findTermsProduct  
вивід termsProduct

**кінець**

*Крок 2*

**початок**

termsProduct =  
findTermsProduct(FIRST\_TERM,  
LAST\_TERM,  
COMMON\_RATIO)  
вивід termsProduct

**кінець**

### Функція findNthTerm(ціле n, дійсний firstTerm, дійсний commonRatio):

*Крок 1*

**початок**

знаходження N-ого елементу  
геометричної прогресії за  
допомогою рекурсії

повернути nthTerm

**кінець**

*Крок 2*

**початок**

**якщо** n != 1  
**то**  
nthTerm =  
findNthTerm(n - 1,  
firstTerm, commonRatio)

**інакше**

nthTerm = firstTerm

**все якщо**

повернути nthTerm

**кінець**

**Функція findTermsProduct(дійсний firstTerm, дійсний lastTerm, дійсний commonRatio):**

*Крок 1*

**початок**

ініціалізації змінних, що  
використовуватимуться при  
роботі функції

знаходження добутку елементів  
геометричної прогресії з першого  
по останній за допомогою  
ітераційного циклу з постумовою  
**повернути** termsProduct

**кінець**

*Крок 2*

**початок**

termsProduct = 1

termIndex = 1

знаходження добутку елементів  
геометричної прогресії з першого  
по останній за допомогою  
ітераційного циклу з постумовою  
**повернути** termsProduct

**кінець**

*Крок 3*

**початок**

termsProduct = 1

termIndex = 1

**повторити**

currentTerm = findNthTerm(termIndex, firstTerm, commonRatio)

termsProduct \*= currentTerm

termIndex += 1

**поки** currentTerm != lastTerm

**все повторити**

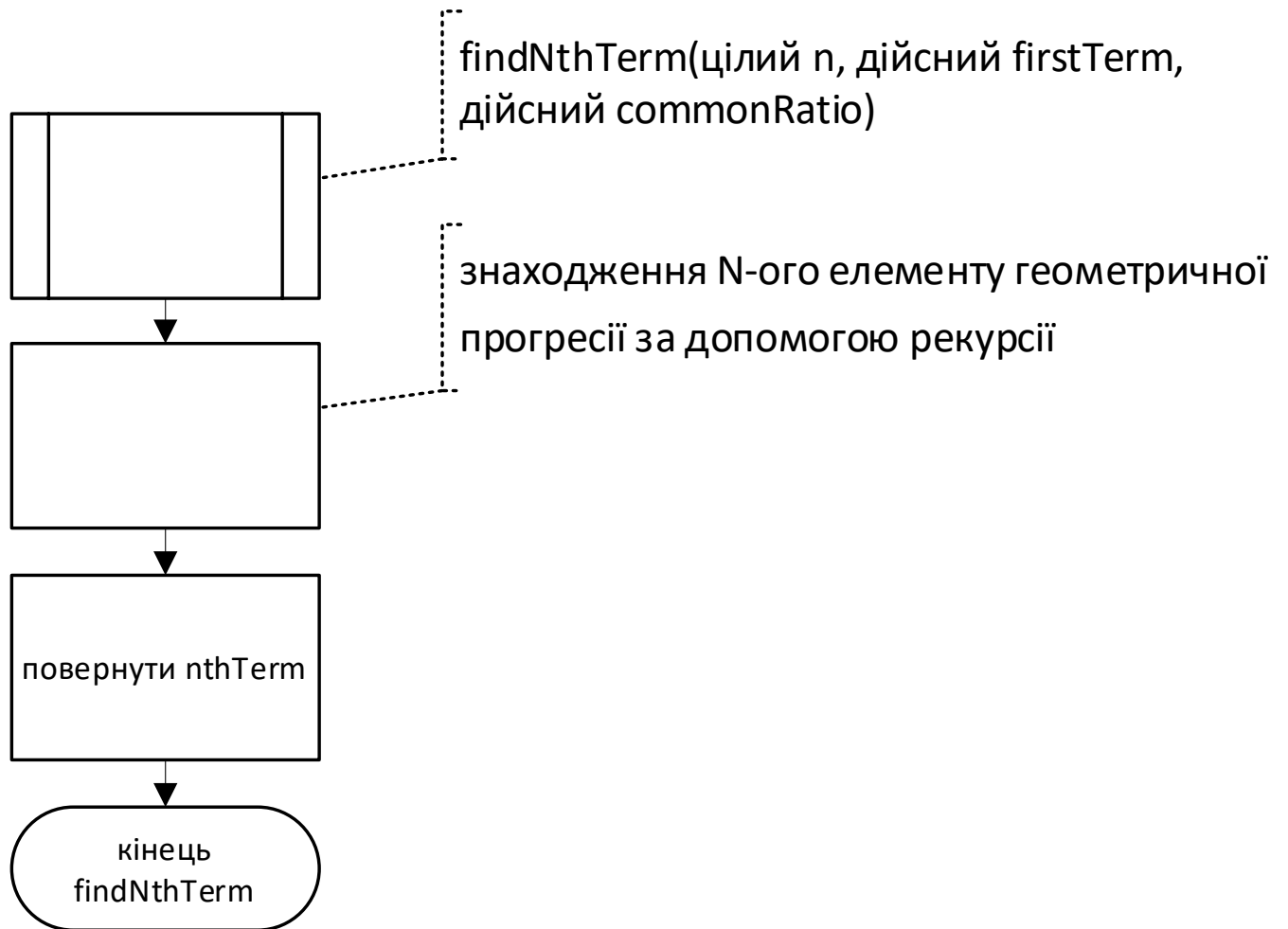
**повернути** termsProduct

**кінець**

**Блок-схема алгоритму.**

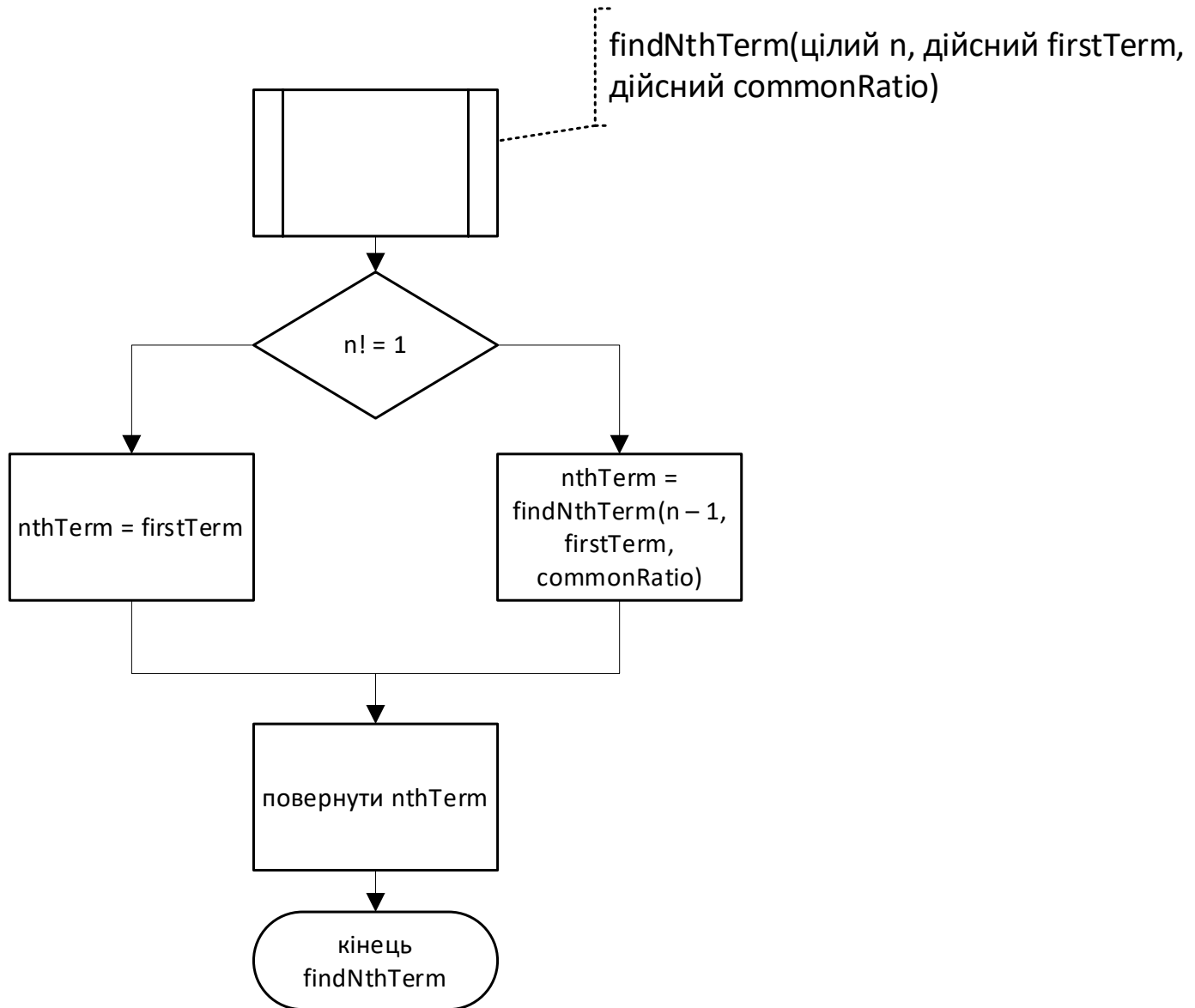
**Функція findNthTerm:**

*Крок 1*



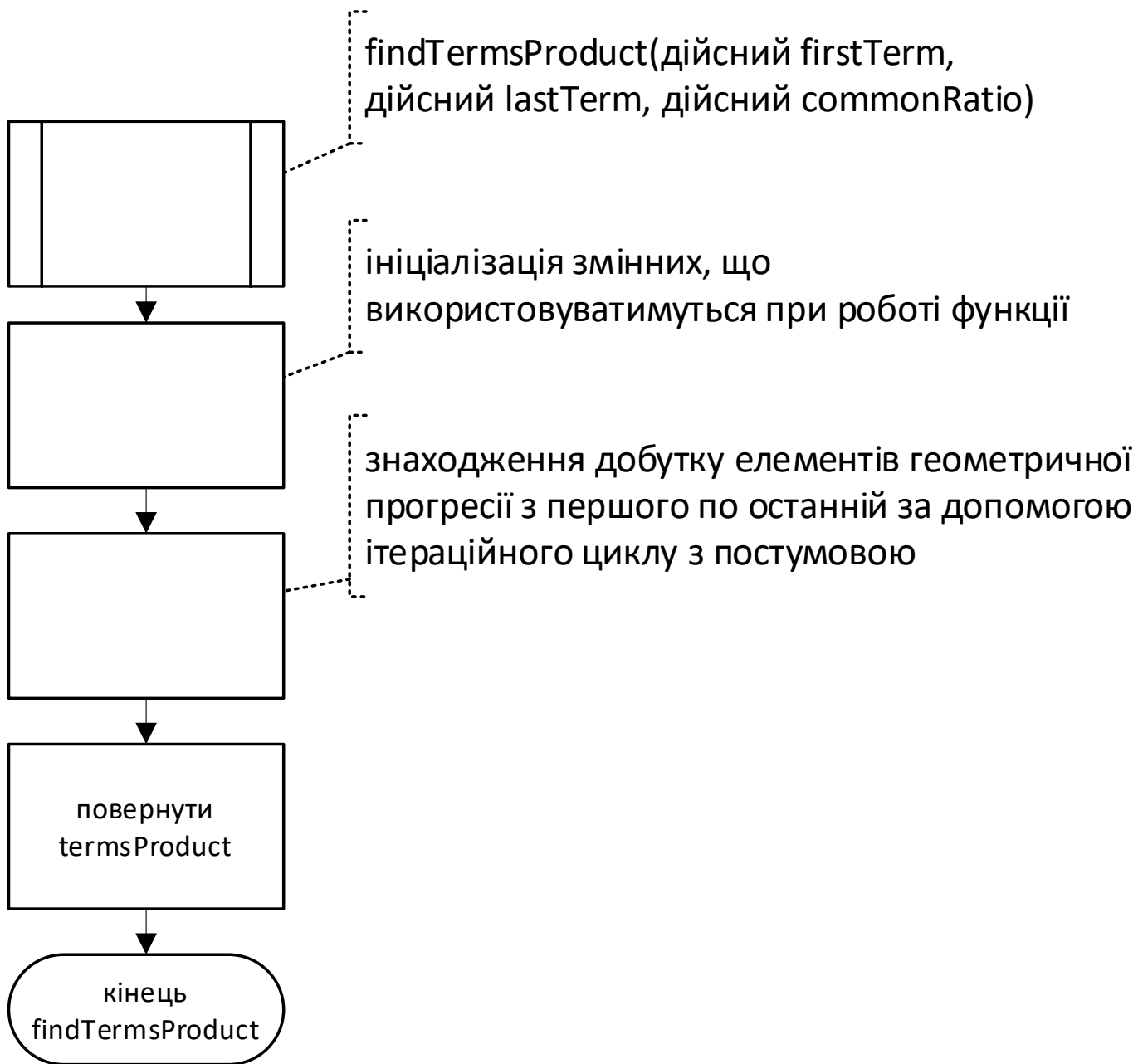


Крок 2

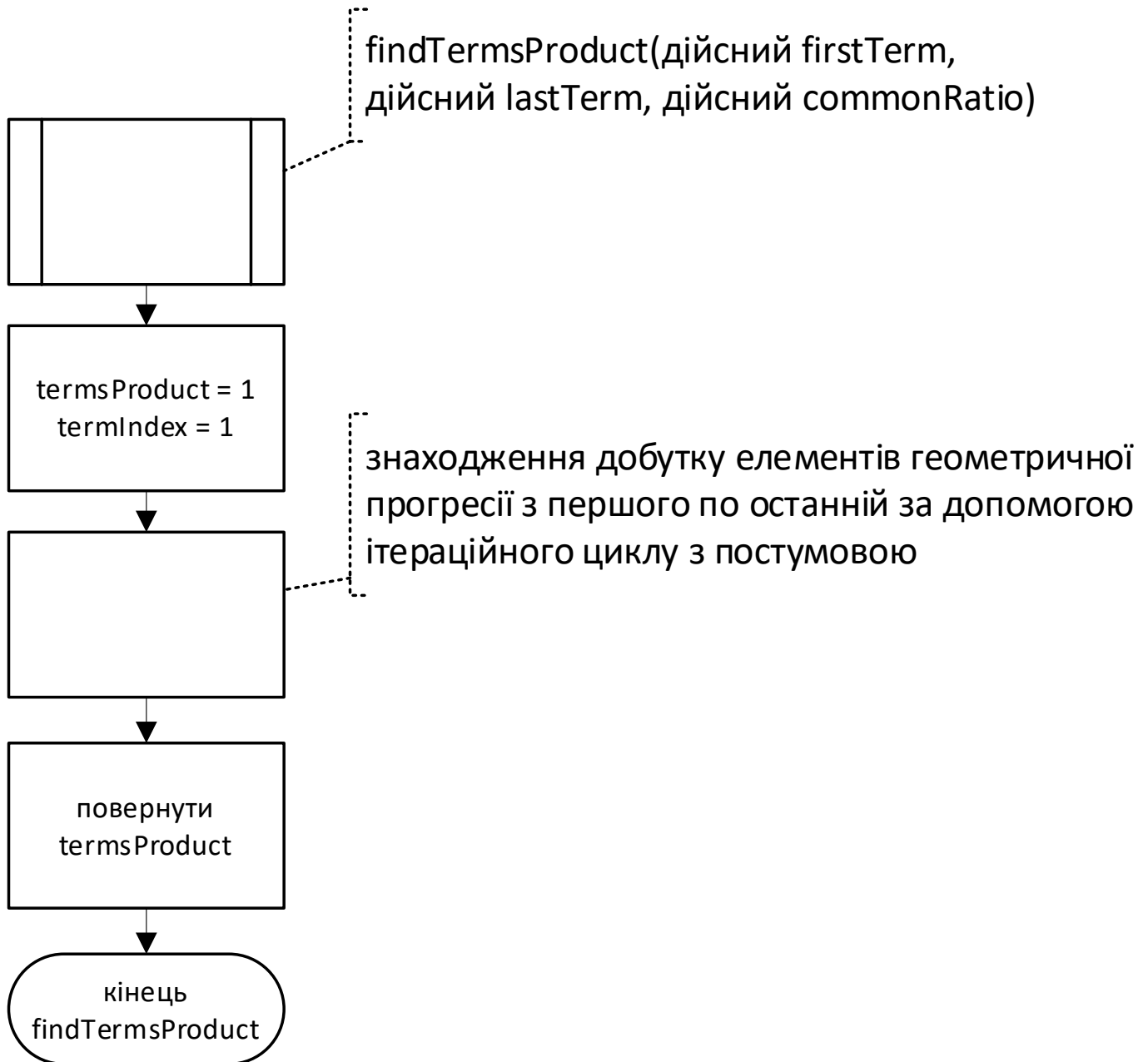


## Функція findTermsProduct:

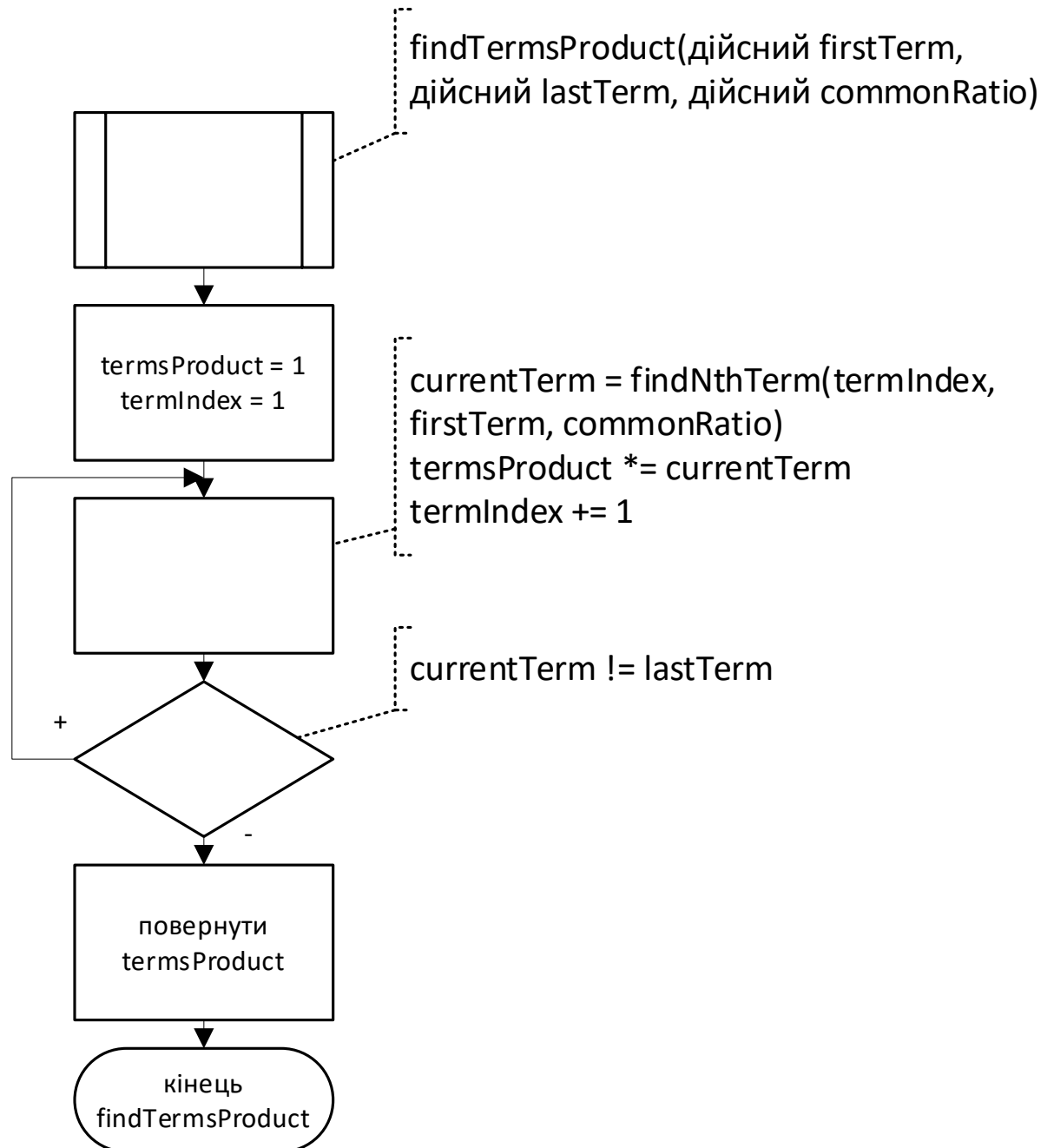
Крок 1



Крок 2

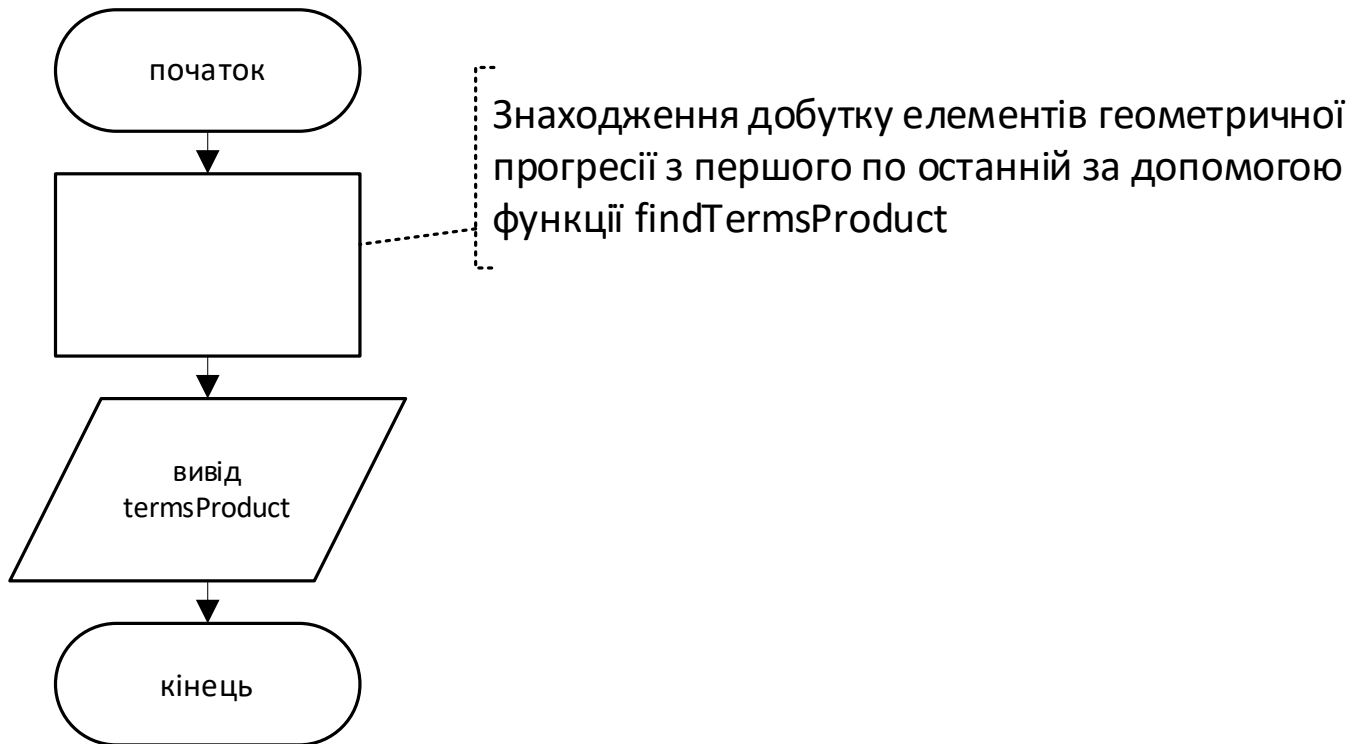


Крок 3

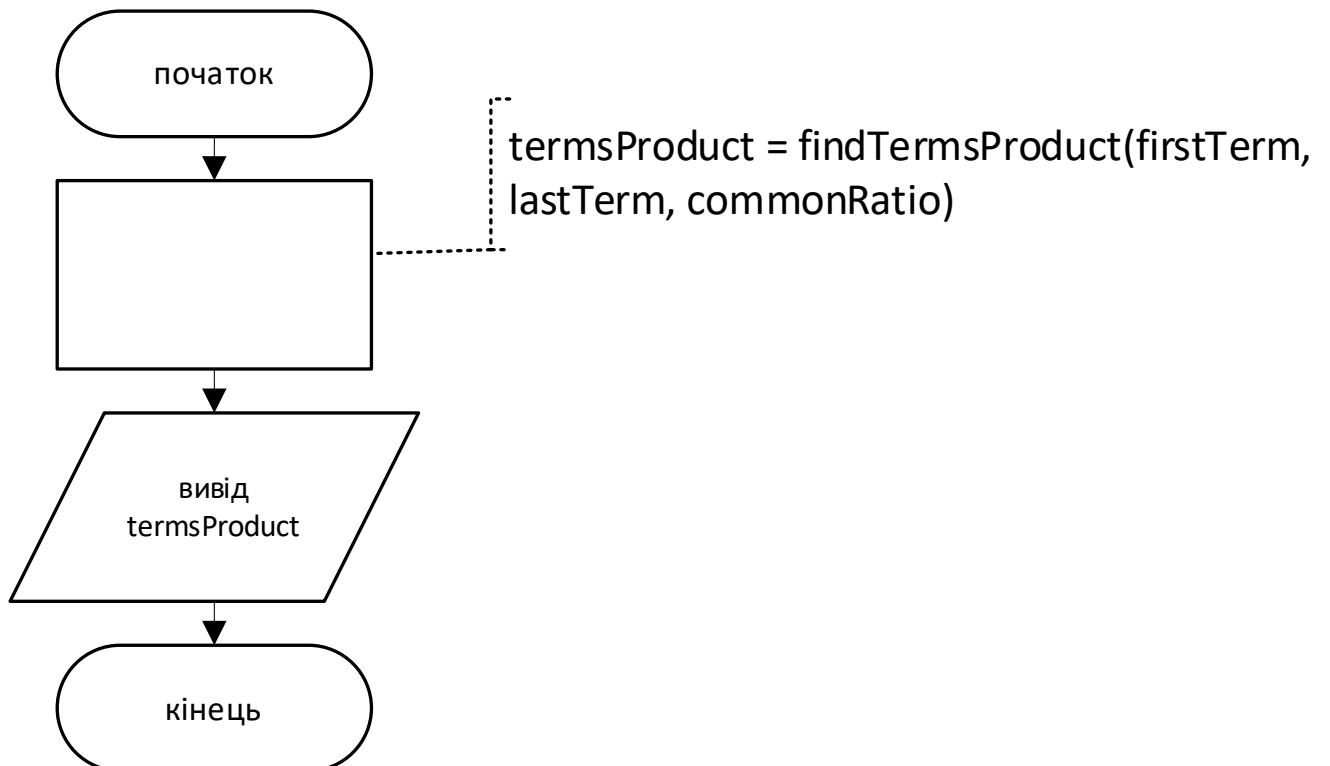


## Основна функція:

### Крок 1



### Крок 2



## Випробування алгоритму.

Приклад знаходження елементу геометричної прогресії з індексом `termIndex = 4`:

Крок	n	nthTerm
1	$4 \neq 1$	$\text{nthTerm} = \text{findNthTerm}(3, 64, 0.25) * 0.25 = 4 * 0.25 = 1$
2	$3 \neq 1$	$\text{nthTerm} = \text{findNthTerm}(2, 64, 0.25) * 0.25 = 16 * 0.25 = 4$
3	$2 \neq 1$	$\text{nthTerm} = \text{findNthTerm}(1, 64, 0.25) * 0.25 = 64 * 0.25 = 16$
4	$1 == 1$	$\text{nthTerm} = 64$

Знаходження інших елементів геометричної прогресії за допомогою функції `findNthTerm` відбувається аналогічно.

$\text{termsProduct} = 64 * 16 * 4 * 1 = 4096$

Вивід: 4096.


## Реалізація алгоритму на мові C++:

```
1 // Variant 2. Program to find the product of terms of the geometric progression
2 // with given first term, last term and common ratio
3
4 #include <iostream>
5
6 // Returns nth term of the geometric progression if n >= 0 and first term != 0 and common ratio != 0
7 // and 0 if not
8 long double findNthTerm(long long n, long double firstTerm, long double commonRatio);
9
10 // Returns the product of terms (from first term to last term) of the geometric progression with known common ratio
11 // if first term != 0 and last term != 0 and common ratio != 0 and 0 if not
12 long double findTermsProduct(long double firstTerm, long double lastTerm, long double commonRatio);
13
14 int main()
15 {
16     const long double FIRST_TERM = 64, LAST_TERM = 1, COMMON_RATIO = 0.25;
17     long double termsProduct;
18     termsProduct = findTermsProduct(FIRST_TERM, LAST_TERM, COMMON_RATIO);
19     std::cout << "The product of terms (from " << FIRST_TERM << " to " << LAST_TERM <<
20         " ) of the geometric progression with common ratio = " << COMMON_RATIO << " is " << termsProduct << "\n";
21
22     return 0;
23 }
24
25 // Returns nth term of the geometric progression if n >= 0 and first term != 0 and common ratio != 0
26 // and 0 if not
27 long double findNthTerm(long long n, long double firstTerm, long double commonRatio)
```

```
28 {
29     long double nthTerm;
30     if ((n >= 1) && firstTerm && commonRatio)
31     {
32         if (n != 1)
33         {
34             nthTerm = findNthTerm(n - 1, firstTerm, commonRatio) * commonRatio;
35         }
36         else
37         {
38             nthTerm = firstTerm;
39         }
40     }
41     else
42     {
43         nthTerm = 0;
44     }
45     return nthTerm;
46 }
47
48 // Returns the product of terms (from first term to last term) of the geometric progression with known common ratio
49 // if first term != 0 and last term != 0 and common ratio != 0 and 0 if not
50 long double findTermsProduct(long double firstTerm, long double lastTerm, long double commonRatio)
51 {
52     long double termsProduct;
53     if (commonRatio && firstTerm && lastTerm)
54     {
55         long double termIndex = 1, currentTerm;
```

```
56         termsProduct = 1;
57         do
58         {
59             currentTerm = findNthTerm(termIndex, firstTerm, commonRatio);
60             termsProduct *= currentTerm;
61             termIndex++;
62         } while (currentTerm != lastTerm);
63     }
64     else
65     {
66         termsProduct = 0;
67     }
68
69     return termsProduct;
70 }
```

## Результат виконання даної програми:



```
Microsoft Visual Studio Debug Console
The product of terms (from 64 to 1) of the geometric progression with common ratio = 0.25 is 4096
C:\Users\user\Desktop\Education\Лабораторні (АСД)\ASD-lab6\Debug\ASD-lab6.exe (process 18308) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

**Висновки.** Таким чином, в результаті виконання лабораторної роботи було досліджено особливості роботи рекурсивних алгоритмів (на прикладі побудови алгоритму для пошуку N-ого елементу геометричної прогресії) та набуто практичних навичок їх використання під час складання програмних специфікацій підпрограм. Також було вивчено особливості опису та використання власних функцій. Особливістю мого варіанту розв’язку завдання було використання вкладеності функції `findNthTerm` у `findTermsProduct`.