Основи програмування – 1. Алгоритми та структури даних

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 № 3 з дисципліни
«Алгоритми та структури даних-1.
Основи алгоритмізації»

«Дослідження ітераційних циклічних алгоритмів»

Варіант 13

Виконав студент ІП-13, Жмайло Дмитро Олександрович
(шифр, прізвище, ім'я, по батькові)

Перевірив _____
(прізвище, ім'я, по батькові)

Київ 2021

Лабораторна робота 3

Дослідження ітераційних циклічних алгоритмів

Мета - дослідити подання операторів повторення дій та набути практичних навичок їх використання під час складання циклічних програмних специфікацій.

Індивідуальне завдання

Варіант 13

13. Для x ∈ [0, 5] з точністю ε = 10^-4 знайти суму парних компонент ряду

$$1-\frac{x^2}{2!}+\frac{x^4}{4!}-\frac{x^6}{6!}+\ldots+(-1)^n\frac{x^{2n}}{(2n)!}\ .$$

Постановка задачі

Необхідно за формулою $1-\frac{x^2}{2!}+\frac{x^4}{4!}-\frac{x^6}{6!}+\ldots+(-1)^n\frac{x^{2n}}{(2n)!}\ .$ знайти суму парних компонент ряду при x ∈ [0, 5] з точністю ε = 10^-4 використовуючи ітераційні цикли.

Побудова математичної моделі

Відповідно до умови складемо таблицю змінних:

Змінна	***Тип***	***Назва***	***Призначення***
Початкове число x	Дійсний	x	Вхідні дані
Задана точність ε	Дійсний	epsilon	Вхідні дані
Номер доданку n	Дійсний	number	Проміжні дані
Лічильник і	Цілий	i	Проміжні дані
Поточна точність	Дійсний	difference	Проміжні дані
Проміжний результат	Дійсний	prev_result	Проміжні дані
Факторіал числа	Цілий	factorial	Проміжні дані
Результат	Дійсний	result	Вихідні дані

Для знаходження значення виразу нам знадобиться використання таких **функцій** :

- **Abs(a),** яка знаходить модуль виразу, де **a -** заданий вираз
- **Pow(a, b),** яка підносить задане число **a** до степеня **b**

Значення змінної **epsilon** є сталим і рівним **10^-4**

Оскільки необхідно знайти суму парних компонент ряду, то для розрахунків будемо використовувати лиш компоненти, **n** яких буде кратний двом. Враховуючи це, початкове значення змінної **number** = 2**,** а кожне наступне значення змінної буде більшим за попереднє на 2. Тому ми можемо не брати до уваги значення виразу **(-1)^n**, оскільки воно завжди буде додатнім і рівним одиниці за означенням.

Для того, щоб увійти в цикл при його першому виконанні, на початку виконання програми прирівняємо значення змінної difference до одиниці

Розв’язання:

Програмні специфікації запишемо у псевдокоді та графічній формі у вигляді блок-схеми.

Крок 1. Визначимо основні дії;

Крок 2. Деталізуємо дію задання значень змінних result, prev_result, difference, epsilon, number;

Крок 3. Деталізуємо дію порівняння значення x;

Крок 4. Деталізуємо дію знаходження суми парних компонент ряду;

Крок 5. Деталізуємо дію знаходження факторіалу числа 4n;

Псевдокод:

Крок 1

початок

введення x
присвоєння початкових значень змінним result, prev_result, difference, epsilon, number
порівняння значення x
знаходження суми парних компонент ряду
знаходження факторіалу числа 4n
виведення result

кінець

Крок 2

початок

введення x
result := 0
prev_result := 0
difference := 1
number := 2
epsilon := Pow(10, -4)
порівняння значення x
знаходження суми парних компонент ряду
знаходження факторіалу числа 4n
виведення result

кінець

Крок 3

початок

введення x
result := 0
prev_result := 0
difference := 1
number := 2
epsilon := Pow(10, -4)
якщо 0≤x && x≤5
то
знаходження суми парних компонент ряду
знаходження факторіалу числа 4n
все якщо
інакше
вивід "Введено некоректне значення змінної x"
виведення result

кінець

Крок 4

початок

```
введення x
result := 0
prev_result := 0
difference := 1
number := 2
epsilon := Pow(10, -4)
якщо 0≤x && x≤5
   то
      повторити
         prev_result := result
         factorial := 1
         i := 2
         знаходження факторіалу числа 2n
         result := prev_result + (Pow(x, number * 2) / factorial)
         difference := Abs(result - prev_result)
         number := number + 2
      поки difference > epsilon
      все повторити
все якщо
інакше
      вивід "Введено некоректне значення змінної x"
виведення result
```

кінець

Крок 5

початок

```
введення x
result := 0
prev_result := 0
difference := 1
number := 2
epsilon := Pow(10, -4)
якщо 0≤x && x≤5
   то
      повторити
         prev_result := result
         factorial := 1
         i := 2
         повторити
            factorial := factorial * i
            i := i + 1
         поки i <= (2 * number)
         все повторити
         result := prev_result + (Pow(x,  number * 2) / factorial)
         difference := Abs(result - prev_result)
         number := number + 2
      поки difference > epsilon
      все повторити
все якщо
інакше
      вивід "Введено некоректне значення змінної x"
виведення result
```

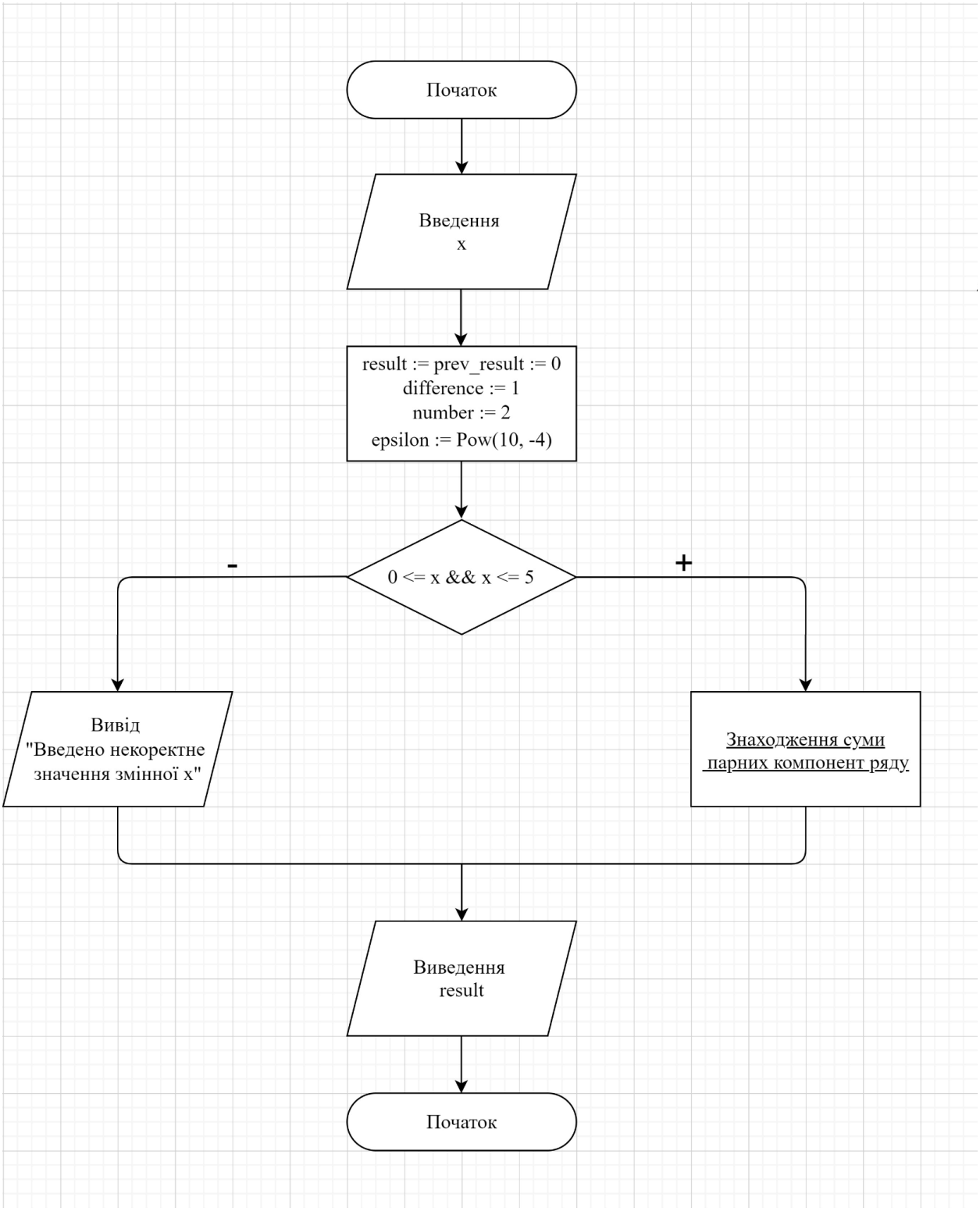
кінець

Блок-схема:

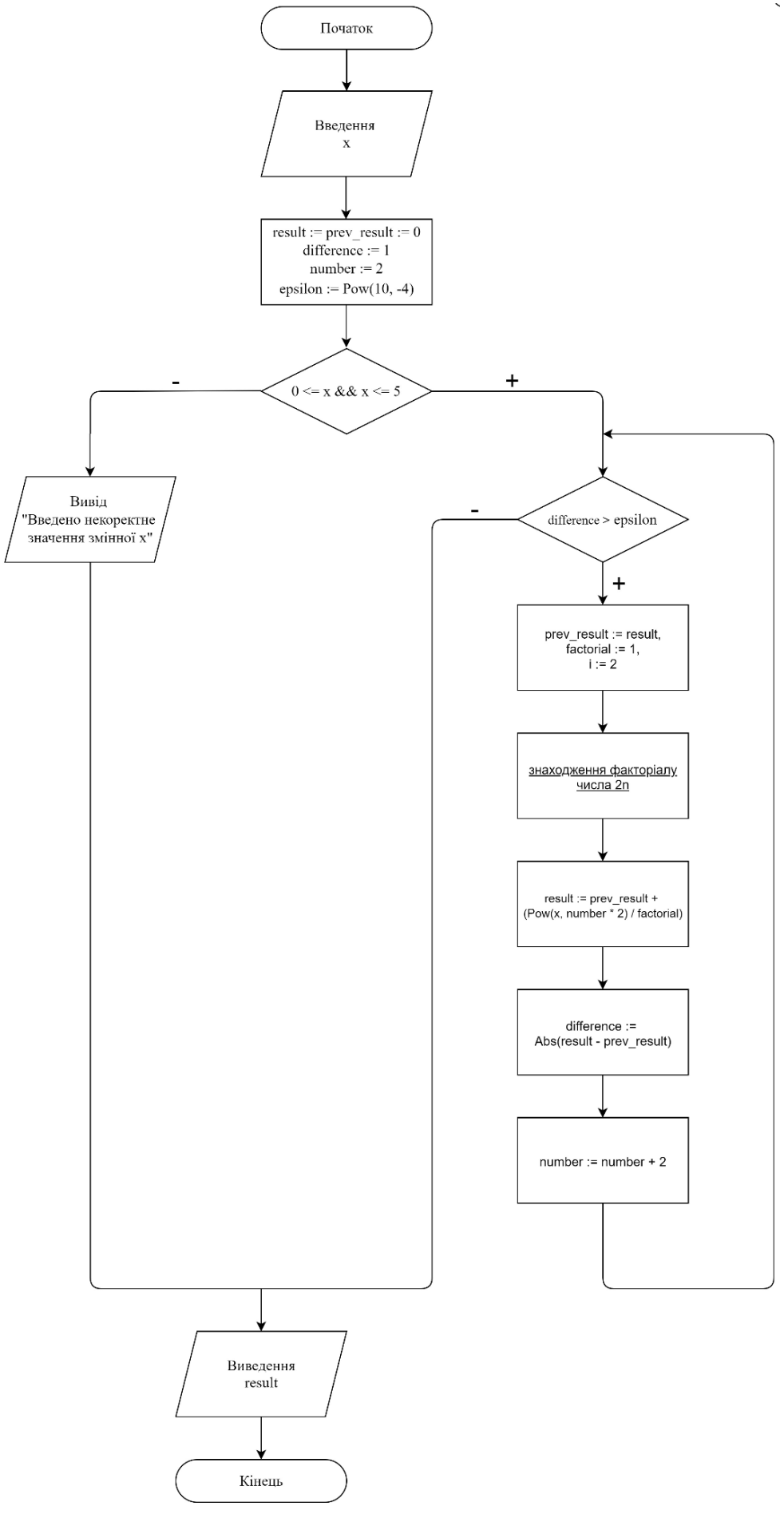
Крок 1

Крок 2

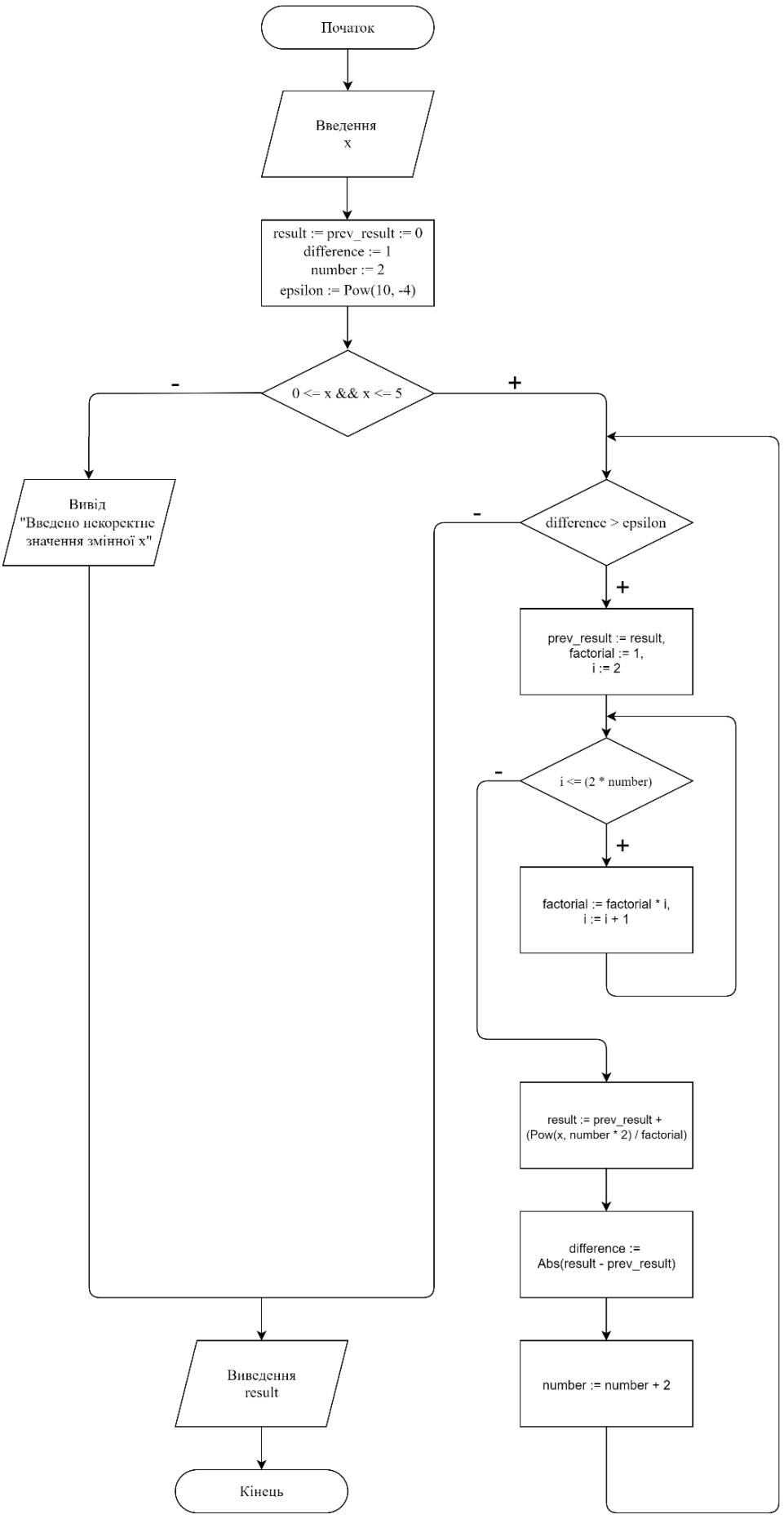
Крок 3


Початок
Введення
x
result := prev_result := 0
difference := 1
number := 2
epsilon := Pow(10, -4)
0 <= x && x <= 5
-
+
Вивід
"Введено некоректне
значення змінної x"
Знаходження суми
парних компонент ряду
Виведення
result
Початок

Крок 4


Початок
Введення
x
result := prev_result := 0
difference := 1
number := 2
epsilon := Pow(10, -4)
0 <= x && x <= 5
-
+
Вивід
"Введено некоректне
значення змінної x"
difference > epsilon
-
+
prev_result := result,
factorial := 1,
i := 2
знаходження факторіалу
числа 2n
result := prev_result +
(Pow(x, number * 2) / factorial)
difference :=
Abs(result - prev_result)
number := number + 2
Виведення
result
Кінець

Крок 5


Початок
Введення
x
result := prev_result := 0
difference := 1
number := 2
epsilon := Pow(10, -4)
0 <= x && x <= 5
-
+
Вивід
"Введено некоректне
значення змінної x"
difference > epsilon
-
+
prev_result := result,
factorial := 1,
i := 2
i <= (2 * number)
-
+
factorial := factorial * i,
i := i + 1
result := prev_result +
(Pow(x, number * 2) / factorial)
difference :=
Abs(result - prev_result)
number := number + 2
Виведення
result
Кінець

- **Випробування алгоритму:**

Блок	**Дія**
	Початок
1	Введення x=-1
2	$0 \leq \boldsymbol{x} \leq 5$ не виконується
3	Вивід: "Введено некоректне значення змінної x"
	Кінець

Блок	**Дія**
	Початок
1	Введення x=2
2	$0 \leq \boldsymbol{2} \leq 5$ виконується
3	**1**>0.0001 виконується
4	number = **2**, factorial = **24,** prev_result = **0,** result **= 0.666666**
5	**0.666666**>0.0001 виконується
6	number = **4**, factorial = **40320,** prev_result = **0.666666,** result = **0.673015**
7	**0.006349**>0.0001 виконується
8	number = **6**, factorial = **479001600,** prev_result = **0.673015,** result = **0.673024**
9	**0.000009**>0.0001 **не** виконується
10	Вивід: **0.673024**
	Кінець

Блок	**Дія**
	Початок
1	Введення x=1
2	$0 \leq \boldsymbol{1} \leq 5$ виконується
3	**1**>0.0001 виконується
4	number = **2**, factorial = **24,** prev_result = **0,** result **= 0.0416666**
5	**0.0416666**>0.0001 виконується
6	number = **4**, factorial = **40320,** prev_result = **0.0416666,** result = **0.0416914**
7	**0.0000248**>0.0001 **не** виконується
8	Вивід: **0.0416914**
	Кінець

Висновок:

Ми дослідили подання операторів повторення дій та набули практичних навичок їх використання під час складання циклічних програмних специфікацій. Навчилися використовувати циклічні програми для знаходження факторіалу числа, зображувати циклічні програми у вигляді блок-схем та псевдокоду.