

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 15

Виконав студент ІП-15, Костін Вадим Анатолійович
(шифр, прізвище, ім'я, по батькові)

Перевірів Вєчерковська Анастасія Сергіївна
(прізвище, ім'я, по батькові)

Київ 2021

Лабораторна робота 6

Дослідження рекурсивних алгоритмів

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

Варіант 15

Задача

Обчислити добуток елементів арифметичної прогресії, що зростає: початкове значення – 2, кінцеве – 30, крок – 4.

Постановка задачі

За умовою задачі прогресія є зростаючою. В умові задачі вказано початкове значення, кінцеве та крок збільшення. Щоб знайти добуток всіх елементів арифметичної прогресії, треба перелічувати кожен із них та помножувати на добуток попередніх елементів.

Математична модель

Перелічувати елементи можна за допомогою ітерації або рекурсії. У цьому завданні будемо використовувати рекурсію. Для цього потрібно створити підпрограму, в якій буде виконуватись звернення підпрограми до самої, але в цьому зверненні вже будуть інші формальні параметри, а саме замість формального параметру кінцевого значення n в зверненні буде формальний параметр кінцевого значення, зменшеного на крок d .

Змінна	Тип	Ім'я	Призначення
Початкове значення	Цілочисельне	A	Фактичний параметр
Кінцеве значення	Цілочисельне	N	Фактичний параметр
Крок	Цілочисельне	D	Фактичний параметр
Добуток елементів	Цілочисельне	S	Результат

Формальний параметр початкового значення	Цілочисельне	a	Формальний параметр
Формальний параметр кінцевого значення	Цілочисельне	n	Формальний параметр
Формальний параметр кроку	Цілочисельне	d	Формальний параметр
Добуток елементів у підпрограмі	Цілочисельне	s	Результат підпрограми

Для виразу $x = x * y$ будемо використовувати $x *= y$

Для перевірки на рівність будемо використовувати логічні вирази $==$, $!=$, $>$, $<$

Крок 1 Деталізуємо основні дії

Крок 2 Деталізуємо дію знаходження добутку елементів арифметичної прогресії, опис термінальної та рекурсивної гілки та повернення результату у підпрограмі

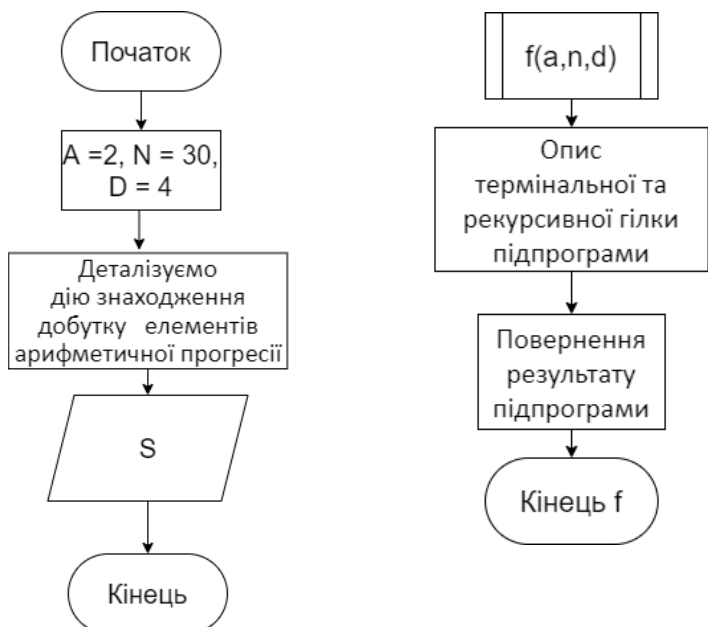
Псевдокод

<p><i>Крок 1</i></p> <p>Початок</p> <p>Ініціалізування A,N,D</p> <p>Деталізуємо дію знаходження добутку елементів арифметичної прогресії</p> <p>Виведення S</p> <p>Кінець</p>	<p><i>Підпрограма $f(a,n,d)$</i></p> <p>Початок</p> <p>Опис термінальної та рекурсивної гілки підпрограми</p> <p>Повернення результату підпрограми</p> <p>Кінець</p>
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

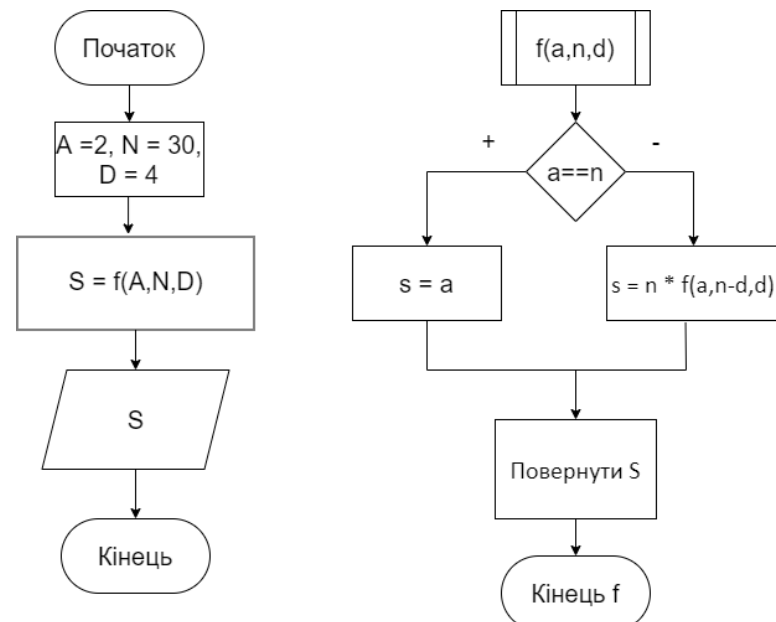
<p>Крок 2</p> <p>Початок</p> <p>Ініціалізування A,N,D</p> <p>$S = f(A,N,D)$</p> <p>Виведення S</p> <p>Кінець</p>	<p>Підпрограма $f(a,n,d)$</p> <p>Початок</p> <p>Якщо $a == n$</p> <p>$s = a$</p> <p>Інакше</p> <p>$s = n * f(a,n-d,d)$</p> <p>Все якщо</p> <p>повернути s</p> <p>Кінець</p>
--------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Блок-схеми

Крок 1



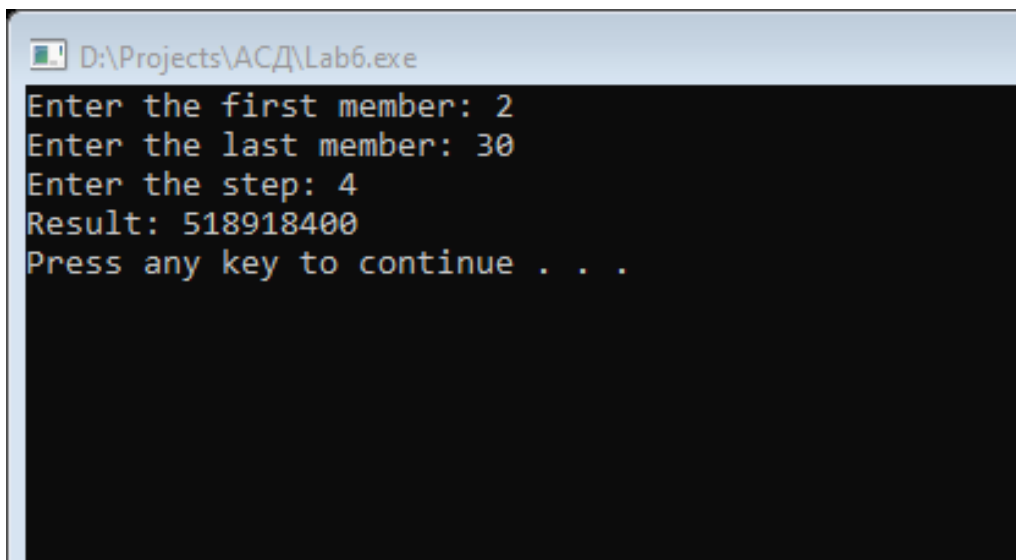
Крок 2



Код програми

```
Lab6.cpp > main()
1  #include <iostream>
2
3  using namespace std;
4
5  int f(int a, int n, int d){
6      int s;
7      if (a==n)
8          s = a;
9      else
10         s = n * f(a,n-d,d);
11     return s;
12 }
13
14 int main(){
15     int A = 2, N = 30, D = 4, S;
16     cout << "The first member: " << A << endl;
17     cout << "The last member: " << N << endl;
18     cout << "The step: " << D << endl;
19     S = f(A,N,D);
20     cout << "Result: " << S << endl;
21     system("pause");
22 }
```

Результат роботи програми



```
D:\Projects\ACД\Lab6.exe
Enter the first member: 2
Enter the last member: 30
Enter the step: 4
Result: 518918400
Press any key to continue . . .
```

Висновки

Протягом шостої лабораторної роботи ми дослідили особливості роботи рекурсивних алгоритмів та набули практичних навичок їх використання під час складання програмних специфікацій. Основними перевагами методу рекурсії над ітерацією є легкість в написанні, а недоліком – довгий час виконання програми.