Алгоритми та структури даних. Основи алгоритмізації

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. Основи алгоритмізації»

«Дослідження складних циклічних алгоритмів»

Варіант 28

Виконав студент: ІП-15 Рибаков Дмитро Вадимович

Перевірив: Вечерковська Анастасія Сергіївна

Київ 2021

Алгоритми та структури даних. Основи алгоритмізації

Лабораторна робота 6

Дослідження рекурсивних алгоритмів

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

Індивідуальне завдання

Варіант 28

Постановка задачі

Розробити рекурсивний алгоритм, псевдокод, блок-схему та код, щоб обчислити значення функції Аккермана для двох невід'ємних цілих чисел n та m, де:

$$A(n,m)=\begin{cases} m+1, \text{ якщо } n=0; \\ A(n-1,1), \text{ якщо } n\neq 0, m=0; \\ A(n-1,A(n,m-1)), \text{ якщо } n>0, m>0. \end{cases}$$

Знайдемо значення функції Аккермана за допомогою використання рекурсивної функції.

Побудова математичної моделі

Змінна	Тип	Ім'я	Призначення
Задане число	Цілий, невід'ємний	n	Вхідні дані
Задане число	Цілий, невід'ємний	m	Вхідні дані
Результат, значення функції Аккермана	Цілий, невід'ємний	result	Вихідні дані

Для знаходження значення функції Аккермана створимо підпрограму - Ackermann(n, m) в якій використаємо умовні оператори з викликами функції Ackermann(n, m), отримане кінцеве значення покладемо у змінну result та виведемо її.

У роботі використовуються наступні дії:

«==» - дорівнює (рівність);

«!=» - не дорівнює (нерівність);

«>» - більше (більше ніж);

«=» - оператор присвоєння;

«&&» - і (логічне множення).

Розв’язання

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

Крок 1. Визначимо основні дії.

Крок 2. Введення двох невід’ємних цілих чисел n та m.

Крок 3. Обчислення значення функції Аккермана.

Крок 3. Виведення результату.

Псевдокод

Основна програма:

крок 1

початок

введення змінних n та m

обчислення значення функції Аккермана

виведення результату

кінець

крок 2

початок

ввід n, m

обчислення значення функції Аккермана

виведення результату

кінець

крок 3

початок

ввід n, m

result = Ackermann(n, m)

виведення результату

кінець

крок 4

початок

ввід n, m

result = Ackermann(n, m)

вивід result

кінець

Підпрограма:

Ackermann(n, m)

якщо n == 0

то повернення m + 1

все якщо

якщо n != 0 && m == 0

то повернення Ackermann(n - 1, 1)

все якщо

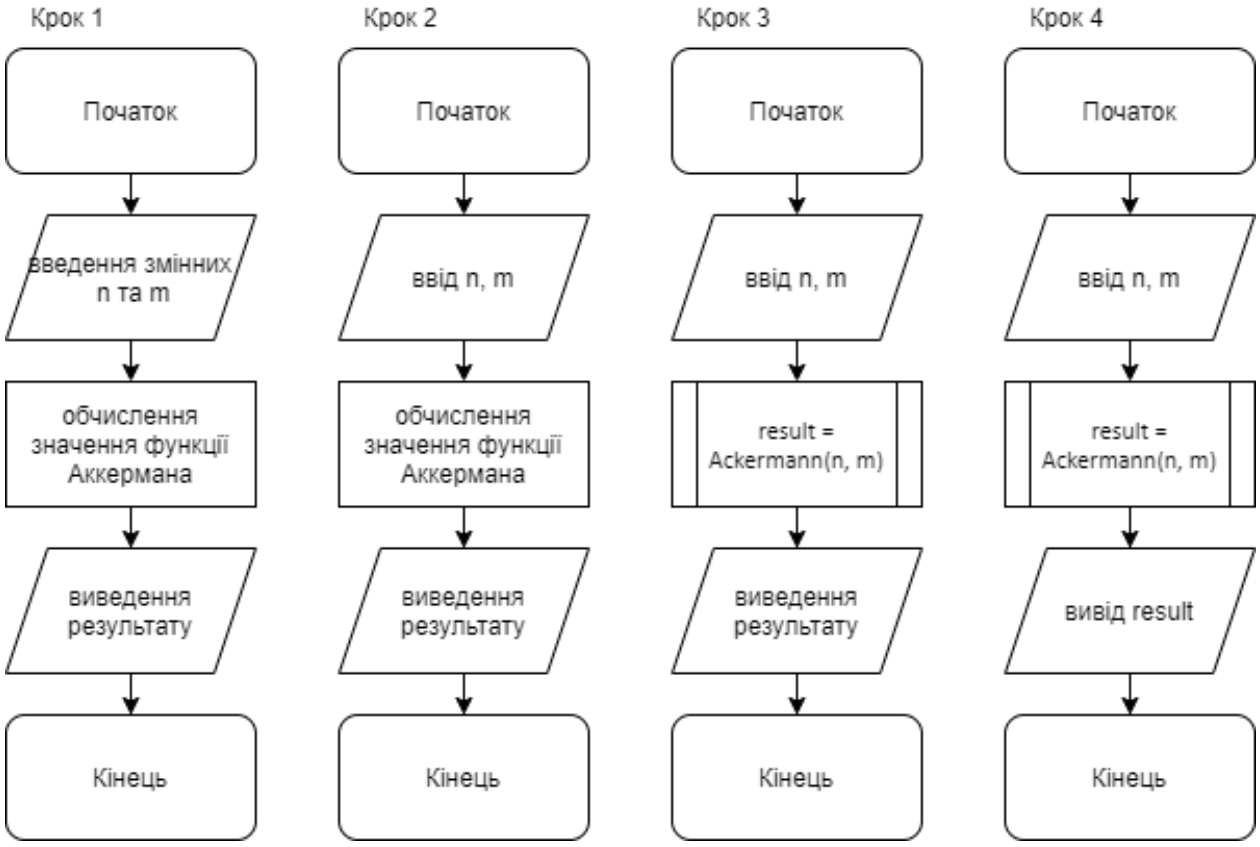
якщо n > 0 && m > 0

то повернення Ackermann(n - 1, Ackermann(n, m - 1))

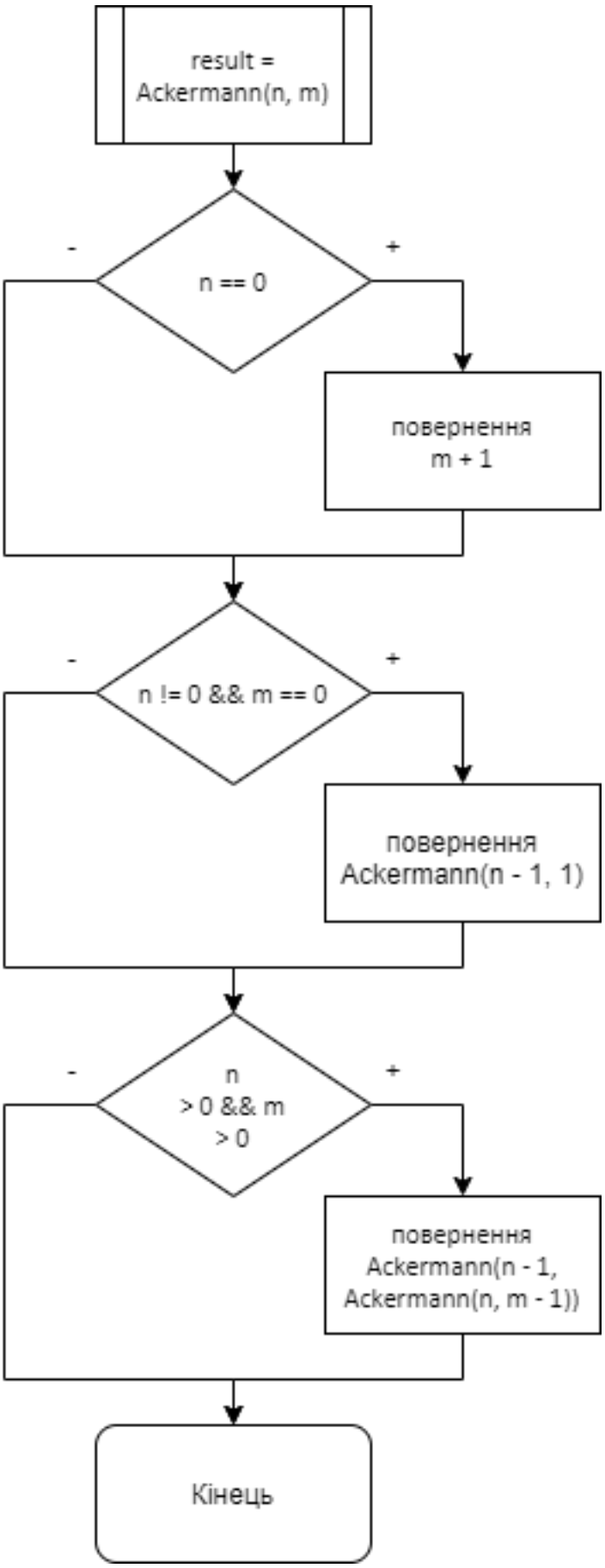
все якщо

Блок-схема

Основна програма:



Підпрограма:



Код програми

```cpp
#include <iostream>
#include <locale.h>

using namespace std;

int Ackermann(int n, int m);

int main()
{
    setlocale(LC_ALL, "Ukrainian");

    cout << "Будь ласка, введіть два невід'ємних цілих числа n та m: ";
    int n, m;
    cin >> n >> m;

        int result;
        result = Ackermann(n, m);
        cout << "Значення функції Аккермана для введених n та m дорівнює: " << result << endl;

    return 0;
}

int Ackermann(int n, int m) {
        if (n == 0) return m + 1;
        if (n != 0 && m == 0) return Ackermann(n - 1, 1);
        if (n > 0 && m > 0) return Ackermann(n - 1, Ackermann(n, m - 1));
}
```

Тестування програми

Блок	Дія
	Початок
1	n = 1, m = 3
2	Ackermann(1 - 1, Ackermann(1, 3 - 1))
3	n = 1, m = 2
4	n = 1, m = 1
5	n = 1, m = 0
6	Ackermann(n - 1, 1)
6	n = 0, m = 1
7	return m + 1 тобто - return 2
8	n = 1, m = 0
9	n = 1, m = 1
10	Ackermann(n - 1, 1)
10	n = 0, m = 2
11	return m + 1 тобто - return 3
12	n = 1, m = 1
13	Ackermann(n - 1, 1)
13	n = 0, m = 3
14	return m + 1 тобто - return 4
15	n = 1, m = 2
16	n = 1, m = 3
17	Ackermann(n - 1, 1)
17	n = 0, m = 4
18	return m + 1 тобто - return 5

19	n = 1, m = 3
20	result = 5
	Кінець

Висновки

На цій лабораторній роботі ми дослідили особливості роботи рекурсивних алгоритмів та набули практичних навичок їх використання під час складання програмних специфікацій підпрограм. В результаті виконання лабораторної роботи ми отримали алгоритм обчислення значення функції Аккермана, при цьому використали рекурсивну функцію, розділили виконання задачи на 4 кроки: визначення основних дій, введення двох невід'ємних цілих чисел n та m, обчислення значення функції Аккермана, виведення результату. Розробили псевдокод блок-схему та код. В процесі випробування ми розглянули один випадок: введення n = 1, m = 3 і виведення результату - result = 5.