Додаток 1

Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Кафедра автоматизованих систем обробки інформації і управління

Звіт

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

Виконав студент ІП-15, Волинець Кирило Михайлович
(шифр, прізвище, ім'я, по батькові)

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

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 6

6.Обчислити добуток елементів арифметичної прогресії, що убуває: початкове значення – 35, кінцеве – 0, крок – 10

Побудова математичної моделі.

Складемо таблицю імен змінних.

Змінна	***Тип***	***Ім'я***	***Призначення***
Початкове значення	Цілочисельний	first_term	Проміжні
Кінцеве значення	Цілочисельний	end	Проміжні
Крок	Цілочисельний	step	Проміжні
Добуток	Цілочисельний	dobutok	Результат
Обчислення	Функція (цілочисельний)	recursion(term)	Обчислення

За умовою перший член прогресії term = 35, крок step = 10, кінцеве значення end = 0, тому ми будемо виконувати рекурсію поки член term більше end. Самі обчислення будуть здійснені у функції recursion(). Там буде задана змінна dobutok що буде прирівняна до поточного члену прогресії term. Якщо член не пересік межу end, то обчислення будуть рекурсивно продовжені, з передаванням у наступну ітерацію поточного члена мінус крок. Після виконання рекурсії добуток буде домножений на її результат. У всіх випадках повертається змінна dobutok.

Розв'язання

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

Крок 1. Визначити основний хід рекурсії

Крок 2. Визначити зупинку рекурсії

Крок 3. Визначити зворотний хід рекурсії

Псевдокод:

початок
    first_term = 35
    step = 10
    end = 0
    виведення recursion(first_term)
кінець

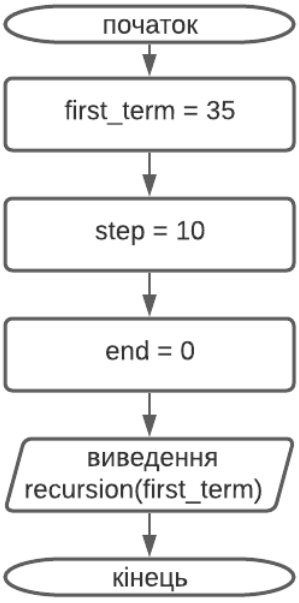
Підпрограми:
Крок 1:
recursion(term)
    dobutok = term
    term = term – step
    dobutok = dobutok * recursion (term)
    Зупинка рекурсії
    Зворотний хід рекурсії
кінець recursion

Крок 2:
recursion(term)
    dobutok = term
    якщо term > end:
        term = term – step
        dobutok = dobutok * recursion (term)
        Зворотний хід рекурсії
    все якщо
    інакше
        Зворотний хід рекурсії
    все інакше
кінець recursion

Крок 3:
recursion(term)
    dobutok = term
    якщо term > end:
        term = term – step
        dobutok = dobutok * recursion (term)

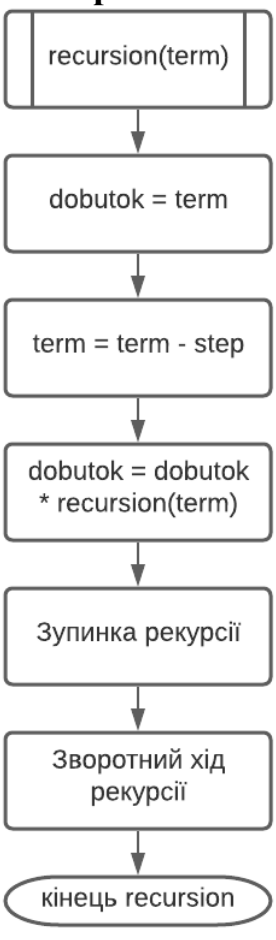
повернути dobutok
все якщо
інакше:
повернути 1
все інакше
кінець recursion

Блок-схема:

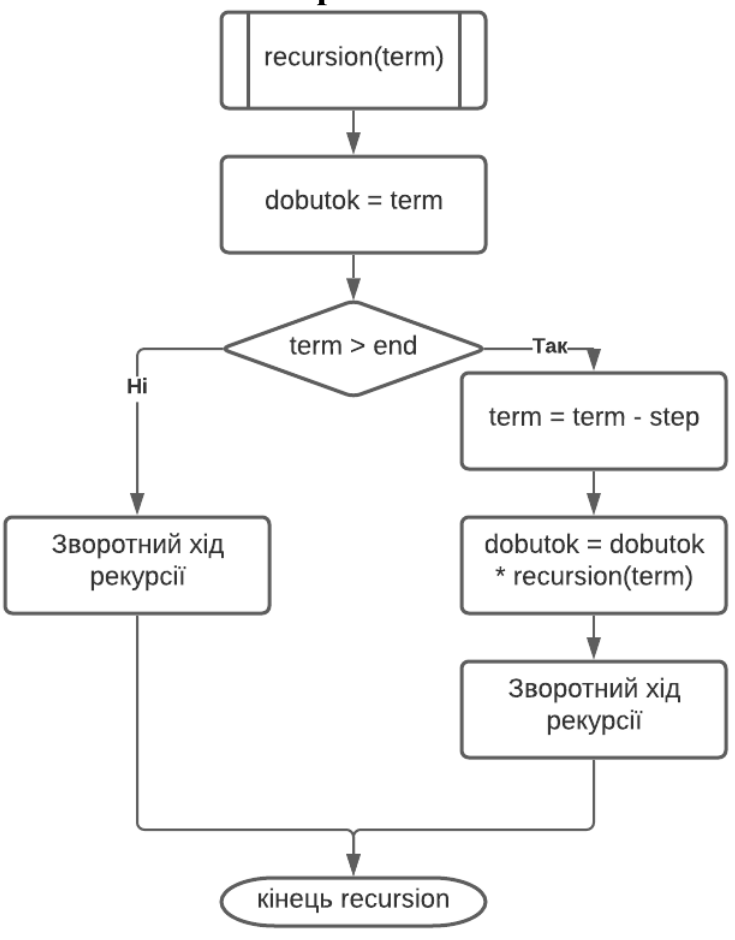


Підпрограми:

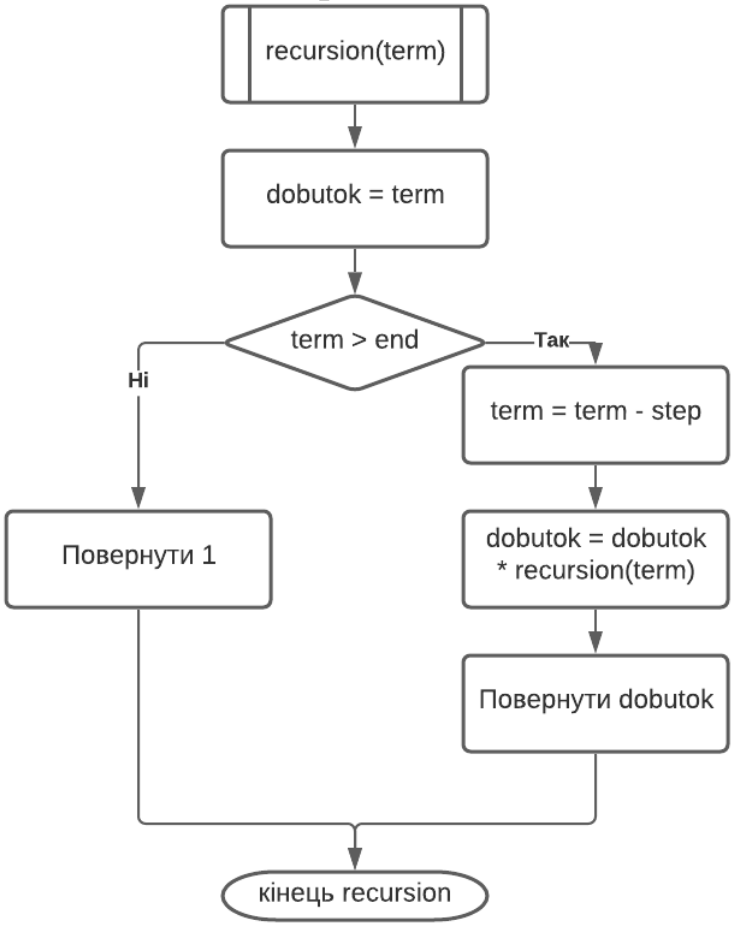
Крок 1:



Крок 2:



Крок 3:


recursion(term)
dobutok = term
term > end
Так
Ні
term = term - step
Повернути 1
dobutok = dobutok
* recursion(term)
Повернути dobutok
кінець recursion

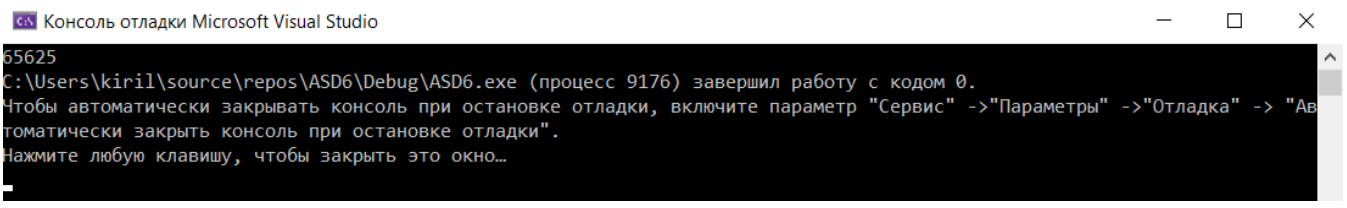
Основи алгоритмізації

Код програми (C++)

```cpp
#include <iostream>
using namespace std;

const int first_term = 35, step = 10, recursion_end = 0;

int recursion(int term) {
    int dobutok = term;
    if (term > recursion_end) {
        term -= step;
        dobutok *= recursion(term);
        return dobutok;
    }
    else return 1;
}

int main()
{
    cout << recursion(first_term);
}
```

Перевірка

	Випадок 1
1	початок
2	first_term = 35
3	step = 10
4	end = 0
5	recursion (35): dobutok = 35 term = 35 – 10 = 25 dobutok = dobutok * recursion (25)
6	recursion (25): dobutok = 25 term = 25 – 10 = 15 dobutok = dobutok * recursion (15)
7	recursion (15): dobutok = 15 term = 15 – 10 = 5 dobutok = dobutok * recursion (5)
8	recursion (5): dobutok = 5 term = 5 – 10 = -5 dobutok = dobutok * recursion (-5)
9	recursion (-5): повернення 1
8	dobutok = dobutok * (-5) = 5 * 1= 5 повернення 5
7	dobutok = dobutok * (-25) = 15 * 5 = 75 повернення 75
6	dobutok = dobutok * 75 = 25 * 75 = 1875 повернення 1875
5	dobutok = dobutok * 1875 = 35 * 1875 = 65625 повернення 65625
4	Вивести 65625
10	кінець



Висновки

Ми дослідили особливості роботи рекурсивних алгоритмів та набули практичних навичок їх використання під час складання програмних специфікацій. В результаті виконання лабораторної роботи ми отримали алгоритм для знаходження добутку елементів арифметичної прогресії, декомпозували рекурсію на 3 кроки: прямий хід, зупинку, та зворотній хід. В процесі випробування ми розглянули випадок з визначеними вхідними даними, та отримали результат 65625, що збігся з виводом реалізації алгоритму на C++.