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 3

Виконав студент ІП-15, Борисик Владислав Тарасович
(шифр, прізвище, ім'я, по батькові)

Перевірила 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 підпрограм.

Варіант 3

Задача

Обчислення суми 10 елементів арифметичної прогресії, що зростає: початкове значення – 10, крок – 2

Постановка задачі

За умовою задачі потрібно суму 10 елементів зростаючої арифметичної прогресії. Початкове значення - 10, крок - 2.

Результатом розв'язку є значення суми елементів арифметичної прогресії.

Побудова математичної моделі

Складемо таблицю змінних

Змінна	Тип	Ім’я	Призначення
Перший елемент прогресії	Цілий	first_element	Початкове дане
Кількість елементів	Цілий	number_of_elements	Початкове дане
Крок	Цілий	step	Початкове дане
Початкове число	Цілий	first_element	Проміжне дане
Кількість елементів	Цілий	number_of_elements	Проміжне дане
Крок	Цілий	step	Проміжне дане
Сума	Цілий	sum	Результат

Для обчислення суми елементів зростаючої арифметичної прогресії будемо використовувати власну рекурсивну функцію recursion(num, count, step). Вона приймає три аргументи:

num - число цілого типу. Початкове число арифметичної прогресії.

count - число цілого типу. Кількість елементів у прогресії.

step - число цілого типу. Крок прогресії.

У тілі функції вона рекурсивно викликає recursion() з аргументами:

num + step, count - 1, step поки count > 0.

Повертає значення суми елементів арифметичної прогресії.

1) Створюємо функцію *recursion*, яка буде приймати 3 аргументи: значення числа, кількості елементів арифметичної прогресії, кроку.

2) Створюємо змінні first_element, number_of_elements, step і присвоюємо ім значення відповідно до умови задачі:
first_element = 10,
number_of_elements = 10,
step = 2.

3) Створюємо змінну *sum* і присвоюємо ій значення суми членів арифметичної прогресії.

4) Виводимо значення змінної *sum*.

Розв'язання

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

Крок 1. Визначимо основні дії

Крок 2. Створення змінних first_element, number_of_elements, step і присвоєння ім значення відповідно до умови задачі:

first_element = 10, number_of_elements = 10, step = 2.

Крок 3. Створення змінної *sum* і присвоєння ій значення суми членів арифметичної прогресії.

Крок 4. Виведення значення змінної *sum*.

Псевдокод

Крок 1

Підпрограма:

Початок

recursion(num, count, step)

якщо count > 0

то

повернути num + recursion(num + step, count - 1, step)

повернути 0

Кінець

Основна програма:

Початок

Створення змінних first_element, number_of_elements, step і присвоєння ім значення: first_element = 10, number_of_elements = 10, step = 2.

Створення змінної *sum* і присвоєння ій значення суми членів арифметичної прогресії.

Виведення *sum*

Кінець

Крок 2

Підпрограма:

Початок

recursion(num, count, step)

якщо count > 0

то

повернути num + recursion(num + step, count - 1, step)

повернути 0

Кінець

Основна програма:

Початок

first_element := 10,

number_of_elements := 10,

step := 2

Створення змінної *sum* і присвоєння ій значення суми членів арифметичної прогресії.

Виведення *sum*

Кінець

Крок 3

Підпрограма:

Початок

recursion(num, count, step)

якщо count > 0

то

повернути num + recursion(num + step, count - 1, step)

повернути 0

Кінець

Основна програма:

Початок

first_element := 10

number_of_elements := 1

step := 2

sum := recursion(first_element, number_of_elements, step)

Виведення *sum*

Кінець

Крок 4

Підпрограма:

Початок

recursion(num, count, step)

якщо count > 0

то

повернути num + recursion(num + step, count - 1, step)

повернути 0

Кінець

Основна програма:

Початок

first_element := 10

number_of_elements := 1

step := 2

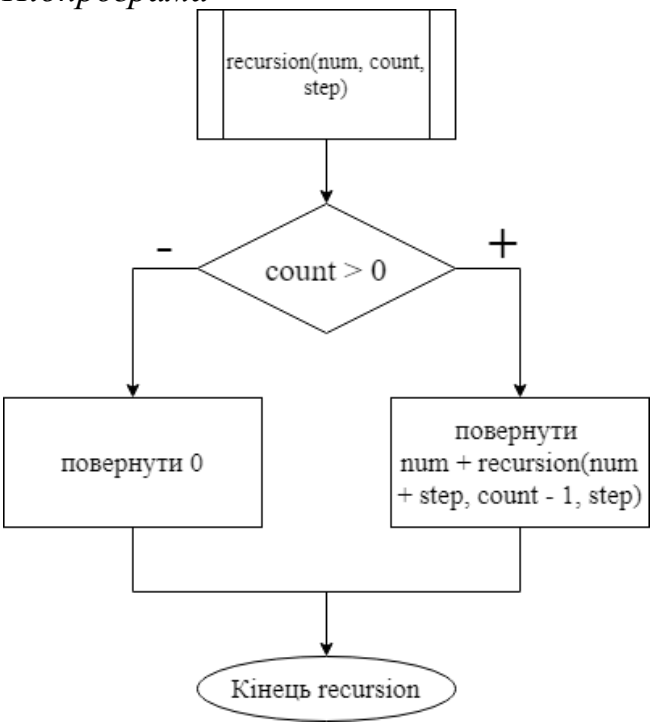
sum := recursion(first_element, number_of_elements, step)

Виведення *sum*

Кінець

Блок-схема алгоритму

Підпрограма

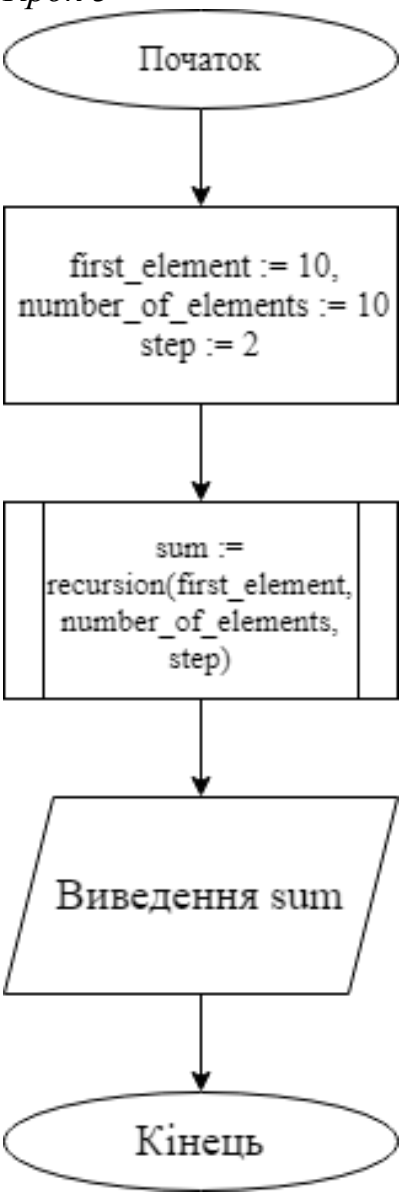


Крок 1

Крок 2


Початок
first_element := 10,
number_of_elements := 10
step := 2
Створення змінної sum і присвоєння їй значення суми членів арифметичної прогресії.
Виведення sum
Кінець

Крок 3


Початок
first_element := 10,
number_of_elements := 10
step := 2
sum := recursion(first_element, number_of_elements, step)
Виведення sum
Кінець

Код

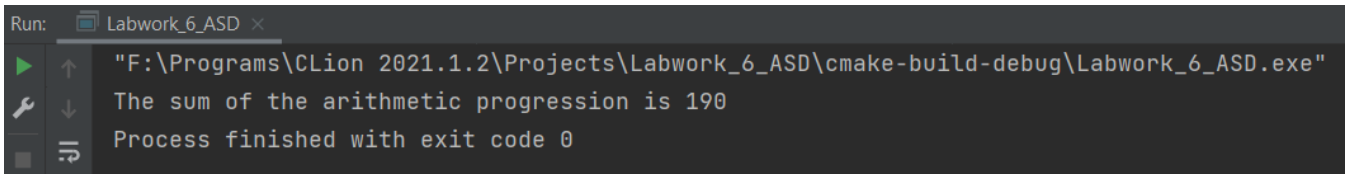
```cpp
#include <iostream>
using namespace std;

/**
    Приймає значення числа, кількості елементів арифметичної прогресії, кроку.
    Повертає суму членів арифметичної прогресії.
*/
int recursion(int num, int count, int step);


int main() {
    const int first_element = 10;
    const int number_of_elements = 10;
    const int step = 2;
    // рахуємо суму членів арифметичної прогресії
    /*  Початкове число:     10
        Кількість елементів: 10
        Крок:                 2    */
    int sum = recursion(first_element,number_of_elements,step);
    printf("The sum of the arithmetic progression is %d", sum);
}


int recursion(int num, int count, int step){
    //якщо кількість елементів більше 0
    if (count > 0)
        /*  Рахуємо значення прогресії:
            число + рекурсивна функція з аргументами:
                число + крок
                кількість елементів - 1
                крок                                    */
        return num + recursion( num: num + step,  count: count - 1, step);
    return 0;
}
```

Результат виконання програми



Випробування алгоритму

Блок	Дія
	Початок
1	first_element = 10
2	number_of_elements = 10
3	step = 2
4	sum = 10
5	sum = 22
6	sum = 36
7	sum = 52
...	
13	sum = 190
14	Виведення: 190
	Кінець

Висновок

Протягом шостої лабораторної роботи я дослідив особливості роботи рекурсивних алгоритмів та набув практичних навичок їх використання під час складання програмних специфікацій підпрограм. В результаті виконання роботи я отримав алгоритм, який рекурсивно знаходить суму елементів арифметичної прогресії, що зростає.