

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний аерокосмічний університет ім. М. Є. Жуковського  
«Харківський авіаційний інститут»

Факультет систем управління літальних апаратів  
Кафедра систем управління літальних апаратів

## Лабораторна робота № 5

з дисципліни «Алгоритмізація та програмування»  
на тему «Реалізація циклічних алгоритмів мовою C ++»

ХАІ.301. спец. група. номер в списку ЛР

Виконав студент гр. \_\_\_\_\_312\_\_\_\_\_

\_\_\_\_\_Ксенія ВЕЛІКОДАНОВА\_\_\_\_\_

(підпис, дата)

(П.І.Б.)

Перевірив

\_\_\_\_\_к.т.н., доц. Олена ГАВРИЛЕНКО

(підпис, дата)

(П.І.Б.)

## МЕТА РОБОТИ

Вивчити теоретичний матеріал із синтаксису мовою C++ і поданням у вигляді UML діаграм циклічних алгоритмів і реалізувати алгоритми з використанням інструкцій циклу з передумовою, циклу з післяумовою і параметризованого циклу мовою C++ в середовищі Visual Studio.

## ПОСТАНОВКА ЗАДАЧІ

Завдання 1. Дано дійсні числа  $(x_i, y_i)$ ,  $i = 1, 2, \dots, n$ , – координати точок на площині. Визначити кількість точок, що потрапляють в фігуру заданого кольору (або групу фігур).

Завдання 2. Дано дійсне число  $x$  і натуральне число  $n$ . Необхідно:

а) Обчислити значення виразу при заданих  $x$  і  $n$  для виразу з табл.2.

б) Вивести: для парних варіантів – значення кожного третього елемента, для непарних – значення кожного четвертого елемента.

Завдання 3. Дослідити ряд на збіжність. Умова закінчення циклу обчислення суми прийняти у вигляді:  $|u_n| < \epsilon$  або  $|u_n| > g$ , де  $\epsilon$  – мала величина для переривання циклу обчислення суми збіжного ряду ( $\epsilon = 10^{-5} \dots 10^{-20}$ );  $g$  – величина для переривання циклу обчислення суми розбіжного ряду ( $g = 10^2 \dots 10^5$ ).

Завдання 4. Організувати меню в командному вікні для багаторазового виконання завдань \*та для перевірки вхідних даних на коректність описати функції, що повертають логічне значення (true – в разі коректного значення переданих параметрів і false – в іншому випадку)

## ВИКОНАННЯ РОБОТИ

Завдання 1.

Вхідні дані (ім'я, опис, тип, обмеження):

$a$  – сторона квадрата, дійсний тип,  $a \geq 0$

$r$  – радіус кола, дійсний тип,  $r \geq 0$ ,  $r < a/2$

$x$  –  $x$  координата точки, дійсний тип

$y$  –  $y$  координата точки, дійсний тип

$k$  – кількість точок, дійсний тип

$crcl$  – рівняння кола для перевірки умови, дійсний тип

$i$  – лічильник точок, цілий тип

Вихідні дані (ім'я, опис, тип):

"Out of region!" – якщо точка не попадає в фігуру

"In region!" – якщо точка попадає в фігуру

"Must be numeric!" – якщо  $x$  і  $y$  не числа

"Must be numeric, non-zero,  $r < a/2$ !" – якщо ввели некоректне значення

Алгоритм вирішення показано на рис. 1

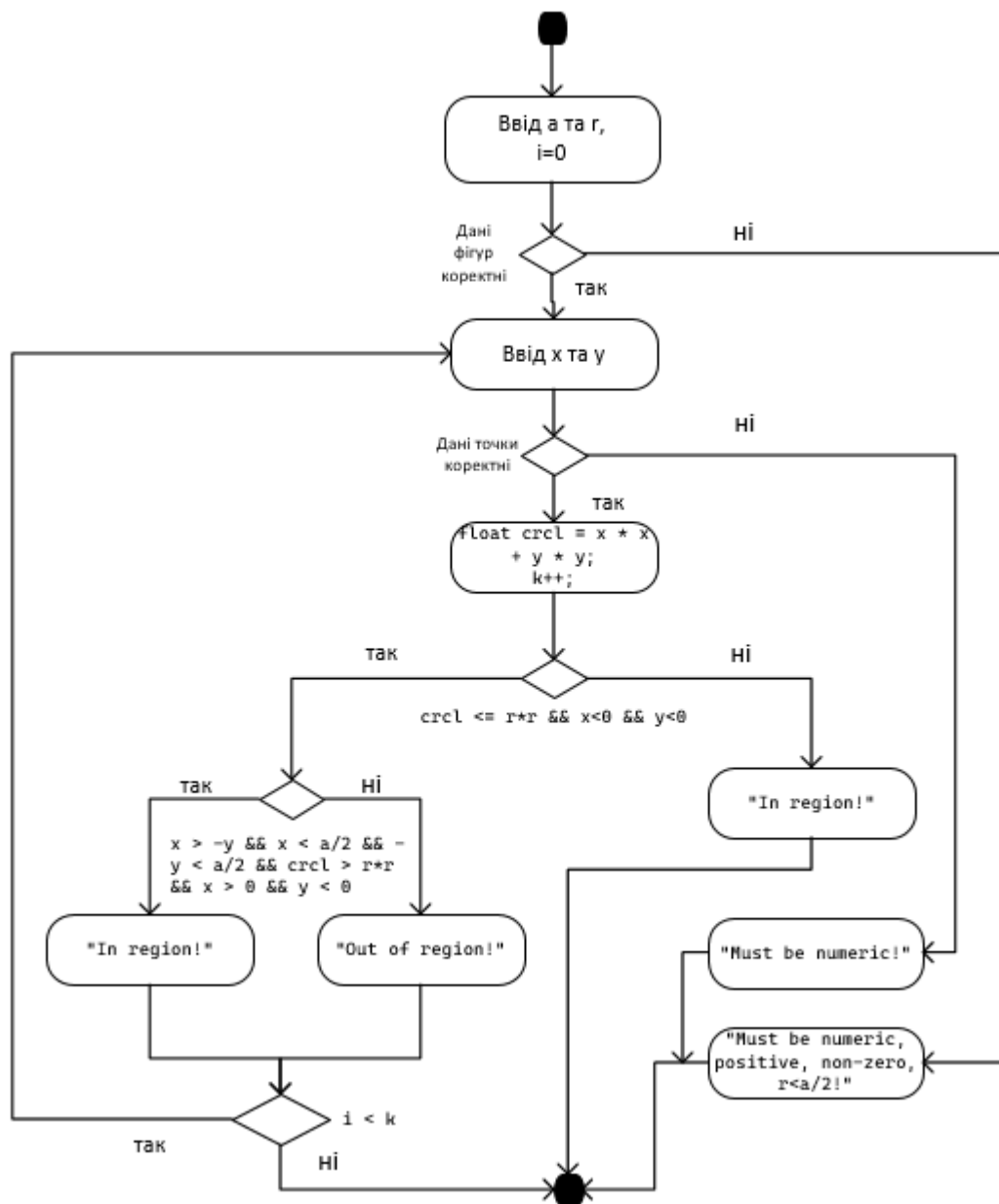


Рисунок 1 – діаграма активності завдання 1

Лістинг коду вирішення наведено в дод. А (стор. 7).

Екрани роботи програми показаний на рис. Б.1. і Б.2

Завдання 2.

Вхідні дані (ім'я, опис, тип, обмеження):

$x$  – змінна, дійсний тип

fk – факторіал суми, дійсний тип

n = кількість повторень суми, дійсний тип

sum = сума числа, дійсний тип

Вихідні дані (ім'я, опис, тип):

"Result: " + sum – кожна третя сума

"Must be number!" – якщо ввели некоректне значення

Лістинг коду вирішення завдання 2 наведено в дод. А (стор. 8).

Екрани роботи програми показаний на рис. Б.3. і Б.4

Завдання 3.

Вхідні дані (ім'я, опис, тип, обмеження):

x - значення x для ряду, дійсний тип

epsilon - умова для збіжності, дійсний тип

g - умова для розбіжності, дійсний тип

M- крок виводу даних, дійсний тип,  $M > 0$

n - початковий індекс, цілий тип

sum - сума ряду, дійсний тип

term- поточний член ряду, дійсний тип

factorial - змінна для обчислення факторіалу в циклі, дійсний тип

numerator - поточний член ряду

Вихідні дані (ім'я, опис, тип):

"n", "u\_n", "term", "sum" - вивід значень з заданим кроком M

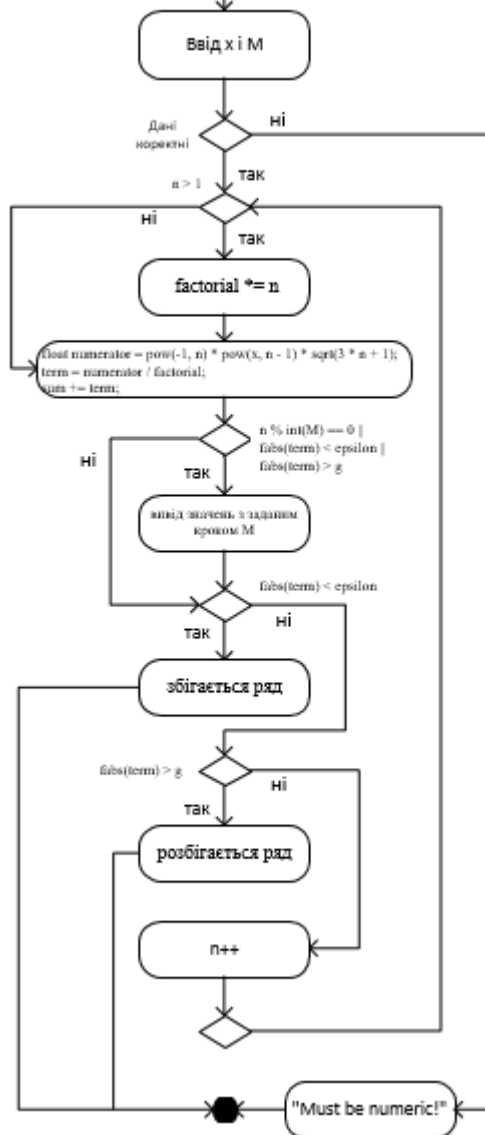
"Suma: " + sum – кожна третя сума

"|u\_n| > g." – збігається ряд

"|u\_n| < epsilon." – розбігається ряд

"Must be numeric or M>0!" – якщо ввели некоректне значення

Алгоритм вирішення показано на рис. 3



Алгоритм вирішення показано на рис. 3

Лістинг коду вирішення завдання 3 наведено в дод. А (стор. 8).

Екрани роботи програми показаний на рис. Б.5. і Б.6

#### Завдання 4.

Вхідні дані (ім'я, опис, тип, обмеження):

menu – вибір завдання, цілий тип

Вихідні дані (ім'я, опис, тип):

"\*\*\*\*\* Geom25\*\*\*\*\*" – перше завдання

"\*\*\*\*\* Suma12\*\*\*\*\*" – друге завдання

"\*\*\*\*\* Suma27\*\*\*\*\*" – третє завдання

"Exit ..." – вихід

"Wrong task! (Only 1,2, 3, -1)" – обрано неіснуюче завдання

Алгоритм вирішення показано на рис. 3

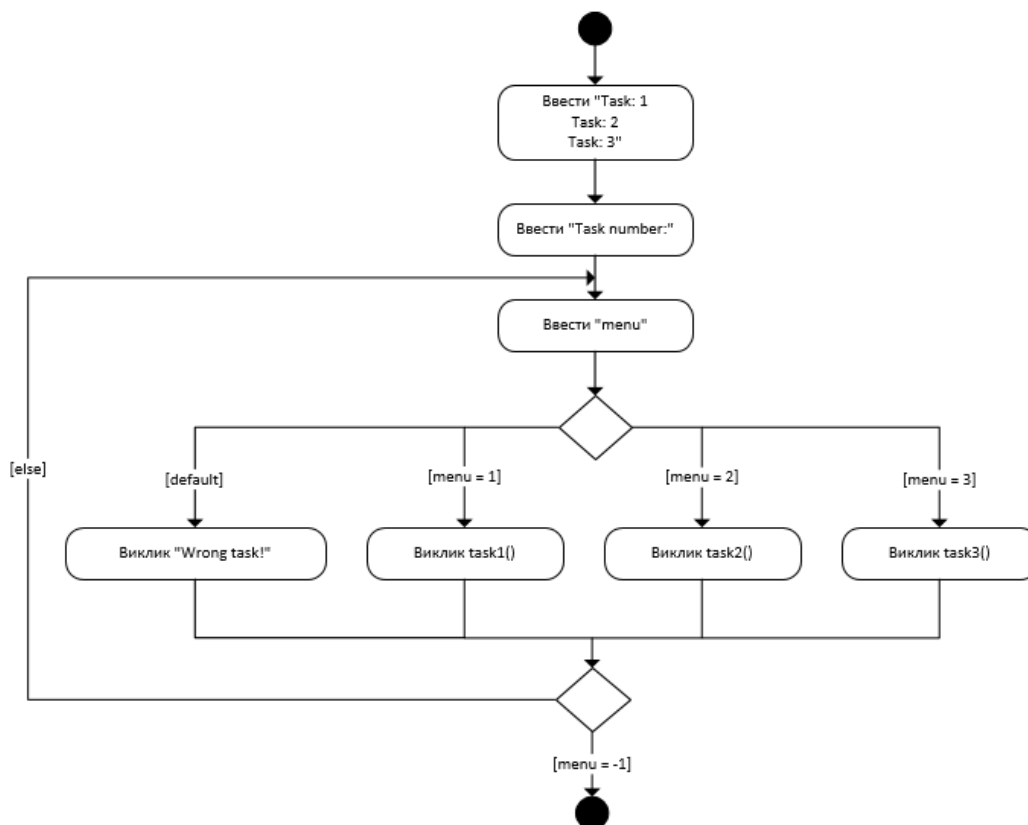


Рисунок 3 – діаграма активності завдання 4

Лістинг коду вирішення завдання 4 наведено в дод. А (стор. 7).

Екрани роботи програми показаний на рис. Б.7. і Б.8

## ВИСНОВКИ

Вивчено теоретичний матеріал із синтаксису мовою C++ і поданням у вигляді UML діаграм циклічних алгоритмів. Закріплено на практиці алгоритми з використанням інструкцій циклу з передумовою, циклу з післяумовою і параметризованого циклу мовою C++ в середовищі Visual Studio.

## ДОДАТОК А

Лістинг коду програми

```

#include <iostream>
using namespace std;

// Дано дійсні числа (xi, yi), i = 1,2, ... n, - координати точок на
// площині.
// Визначити кількість точок, що потрапляють в фігуру заданого
// кольору (або групу фігур).
void task1();

// Дано дійсне число x і натуральне число n. Необхідно:
// а) Обчислити значення виразу при заданих x і n для виразу з табл.2.
// б) Вивести: для парних варіантів - значення кожного третього елемента, для
// непарних - значення кожного четвертого елемента.
void task2();

// Дослідити ряд на збіжність. Умова закінчення циклу обчислення суми прийняти у
// вигляді:
// | un | < e або | un | > g, де e - мала величина для переривання циклу
// обчислення суми збіжного ряду
// (e = 10-5 ... 10-20);
// g - величина для переривання циклу обчислення суми розбіжного ряду (g = 102 ...
// 105).
void task3();

// Перевірка коректності
bool check_in(float &in);

int main() {
    int menu; // Змінна для номера завдання
    cout << "Task1: 1\nTask2: 2\nTask3: 3\nExit: -1" << endl;
    do
    { // Початок циклу
        cout << "Task number:";
        cin >> menu; // Вибір номера завдання

        // Перевірка вводу
        if (cin.fail()) {
            cin.clear(); // Скидає стан помилки
            cin.ignore(10000, '\n'); // Очищає буфер
            continue; // Повертається до початку циклу
        }

        switch (menu) {
            case 1: task1(); break; // Завдання 1
            case 2: task2(); break; // Завдання 2
            case 3: task3(); break; // Завдання 3
            case -1: cout << "Exit ..." << endl; break; // Вихід
            default: cout << "Wrong task!" << endl; // Інший номер - повторити
        }
    } // Кінець циклу
    while (menu != -1); // Умова виконання: поки не -1
    return 0;
}

// Завдання 1
void task1() {

```

```

float x, y, a, r, k;
cout << "***** Geom25*****" << endl;
cout << "Enter a, r, k: " << endl;
// Перевірка коректності даних !!!
if (!check_in(a) || !check_in(r) || !check_in(k)) {
    // Повідомлення про помилку
    cout << "Must be numeric, positive, non-zero, r<a/2!" << endl;
}
else // Дані коректні
{
    for (int i = 0; i < k; i++) {
        // Перевірка коректності даних !!!
        cout << "Enter x, y: " << endl;
        if (check_in(x) || check_in(y)) {
            cout << "Must be numeric!" << endl;
        }
        else
        {
            // Обчислення для кола
            float crcl = x * x + y * y;
            // Чверть кола
            if (crcl <= r * r && x < 0 && y < 0)
                cout << "In region!" << endl;
            // Частка квадрата
            else if (x > -y && x < a / 2 && -y < a / 2 && crcl > r *
r && x > 0 && y < 0)
                cout << "In region!" << endl;
            else cout << "Out of region!" << endl;
        }
    }
}

// Завдання 2
void task2() {
    float x, fk=1, n, sum = 0;
    cout << "***** Suma12*****" << endl;
    cout << "Enter x, n: " << endl;
    // Перевірка коректності даних
    if (!check_in(x) || !check_in(n)) {
        cout << "Must be numeric!" << endl;
    }
    else { // Дані коректні
        for (int k = 0; k < n; k++) {
            for (int i = 1; i <= k; i++) {
                fk = fk * i; // Факторіал k
            }
            sum += pow(x - k, k + 1) / fk; // Сума
            if (int(k) % 3 == 0) { // Кожна третя сума
                cout << "Result: " << sum << endl;
            }
        }
    }
}

// Завдання 3
void task3() {
    float x; // Значення x для ряду
    float epsilon = 1e-20; // Умова для збіжності
    float g = 1e20; // Умова для розбіжності
    float M; // Крок виводу даних
    int n = 1; // Початковий індекс

```



```

float sum = 0.0;           // Сума ряду
float term;               // Поточний член ряду
float factorial = 1.0;     // Змінна для обчислення факторіалу в циклі

cout << "***** Suma27*****" << endl;
cout << "Enter x, M: " << endl;
// Перевірка коректності даних
if (!check_in(x) || !check_in(M) || M<0) {
    cout << "Must be numeric or M>0!" << endl;
}
else { // Дані коректні

    do {
        // Оновлення факторіалу для поточного n
        if (n > 1) factorial *= n;

        // Обчислення поточного члена ряду
        float numerator = pow(-1, n) * pow(x, n - 1) * sqrt(3 * n + 1);
        term = numerator / factorial;

        // Додавання члена до суми
        sum += term;

        // Вивід значень з заданим кроком M
        if (n % int(M) == 0 || fabs(term) < epsilon || fabs(term) > g)
        {
            cout << "n = " << n << ", u_n = " << term << ", sum = "
<< sum << endl;
        }

        // Перевірка умов закінчення циклу
        if (fabs(term) < epsilon) {
            cout << "|u_n| < epsilon." << endl; // Збігається
            break;
        }
        if (fabs(term) > g) {
            cout << "|u_n| > g." << endl; // Розбігається
            break;
        }

        ++n; // Збільшення індексу
    } while (true);

    cout << "Suma: " << sum << endl;
}

// Функція для перевірки помилок консольного введення
bool check_in(float& in) {
    cin >> in;
    if (!cin) { // Перевірка помилок консольного введення
        cin.clear(); // Очистка стану помилки
        cin.ignore(10000, '\n'); // Очищення буфера вводу
        return false;
    }
    else return true;
}

```

ДОДАТОК Б  
Скрін-шоти вікна виконання програми

```
Task1: 1
Task2: 2
Task3: 3
Exit: -1
Task number:1
***** Geom25*****
Enter a, r, k:
20 8 2
Enter x, y:
9 -8
In region!
Enter x, y:
9 9
Out of region!
Task number:
```

Рисунок Б.1 – Екран виконання програми для вирішення завдання 1.1

```
Task1: 1
Task2: 2
Task3: 3
Exit: -1
Task number:1
***** Geom25*****
Enter a, r, k:
t 4
4
Must be numeric, positive, non-zero,  $r < a/2$ !
```

Рисунок Б.2 – Екран виконання програми для вирішення завдання

```
Task1: 1
Task2: 2
Task3: 3
Exit: -1
Task number:2
***** Suma12*****
Enter x, n:
2 2
Result: 2
```

Рисунок Б.3 – Екран виконання програми для вирішення завдання 2.1

```
***** Suma12*****
Enter x, n:
f 2
Must be numeric!
```

Рисунок Б.4 – Екран виконання програми для вирішення завдання  
2.2

Рисунок Б.5 – Екран виконання програми для вирішення завдання 3.1

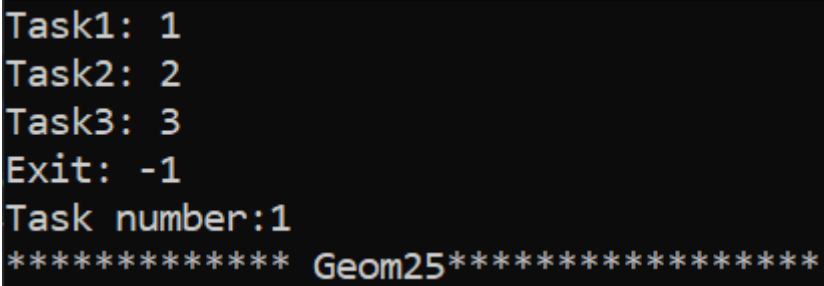
```

***** Suma27*****
Enter x, M:
2 1
n = 1, u_n = -2, sum = -2
n = 2, u_n = 2.64575, sum = 0.645751
n = 3, u_n = -2.10819, sum = -1.46243
n = 4, u_n = 1.20185, sum = -0.260583
n = 5, u_n = -0.533333, sum = -0.793917
n = 6, u_n = 0.193729, sum = -0.600188
n = 7, u_n = -0.0595608, sum = -0.659749
n = 8, u_n = 0.015873, sum = -0.643876
n = 9, u_n = -0.00373298, sum = -0.647609
n = 10, u_n = 0.000785575, sum = -0.646823
n = 11, u_n = -0.000149584, sum = -0.646973
n = 12, u_n = 2.60072e-05, sum = -0.646947
n = 13, u_n = -4.16016e-06, sum = -0.646951
n = 14, u_n = 6.16192e-07, sum = -0.64695
n = 15, u_n = -8.49766e-08, sum = -0.64695
n = 16, u_n = 1.0963e-08, sum = -0.64695
n = 17, u_n = -1.32866e-09, sum = -0.64695
n = 18, u_n = 1.51827e-10, sum = -0.64695
n = 19, u_n = -1.64119e-11, sum = -0.64695
n = 20, u_n = 1.6831e-12, sum = -0.64695
n = 21, u_n = -1.6419e-13, sum = -0.64695
n = 22, u_n = 1.52722e-14, sum = -0.64695
n = 23, u_n = -1.35742e-15, sum = -0.64695
n = 24, u_n = 1.15517e-16, sum = -0.64695
n = 25, u_n = -9.42933e-18, sum = -0.64695
n = 26, u_n = 7.39511e-19, sum = -0.64695
n = 27, u_n = -5.5809e-20, sum = -0.64695
n = 28, u_n = 4.05862e-21, sum = -0.64695
|u_n| < epsilon.
Suma: -0.64695

***** Suma27*****
Enter x, M:
g 8
Must be numeric!

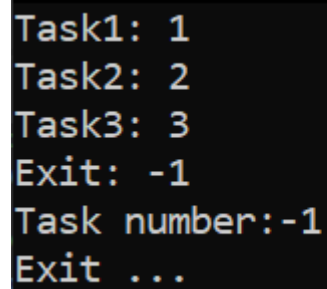
```

Рисунок Б.6 – Екран виконання програми для вирішення завдання



```
Task1: 1
Task2: 2
Task3: 3
Exit: -1
Task number:1
***** Geom25*****
```

Рисунок Б.7 – Екран виконання програми для вирішення завдання 4.1



```
Task1: 1
Task2: 2
Task3: 3
Exit: -1
Task number:-1
Exit ...
```

Рисунок Б.8 – Екран виконання програми для вирішення завдання

4.2