ЗВІТ

про виконання лабораторної роботи № 7.1 рекурсивним способом

«Пошук заданих елементів та впорядкування рядків / стовпчиків матриці»

з дисципліни

«Алгоритмізація та програмування»

студентки групи ІК-11

Турковської Марти Михайлівни

**Умова завдання:**

Написати програму, яка:

1. за допомогою генератора випадкових чисел формує вказану матрицю – так, що значення її елементів належать заданому діапазону. Обчислити кількість та суму тих елементів, які задовольняють вказаному критерію; а також замінити нулями ці елементи;

2. впорядковує матрицю, переставляючи місцями її рядки / стовпчики відповідно до вказаних ключів впорядкування.

Пояснити призначення змінних.

Необхідно:

- (завдання 1 і 2) сформувати матрицю;

- (завдання 1 і 2) вивести її на екран у вигляді таблиці, використовуючи форматне виведення;

- (завдання 2) впорядкувати матрицю, переставляючи місцями її рядки / стовпчики відповідно до вказаних ключів впорядкування;

- (завдання 2) вивести на екран модифіковану матрицю у вигляді окремої таблиці, використовуючи виведення з тими самими специфікаціями формату;

- (завдання 1) виконати вказані у завданні дії (обчислити кількість та суму тих елементів, які задовольняють вказаному критерію; а також замінити нулями ці елементи);

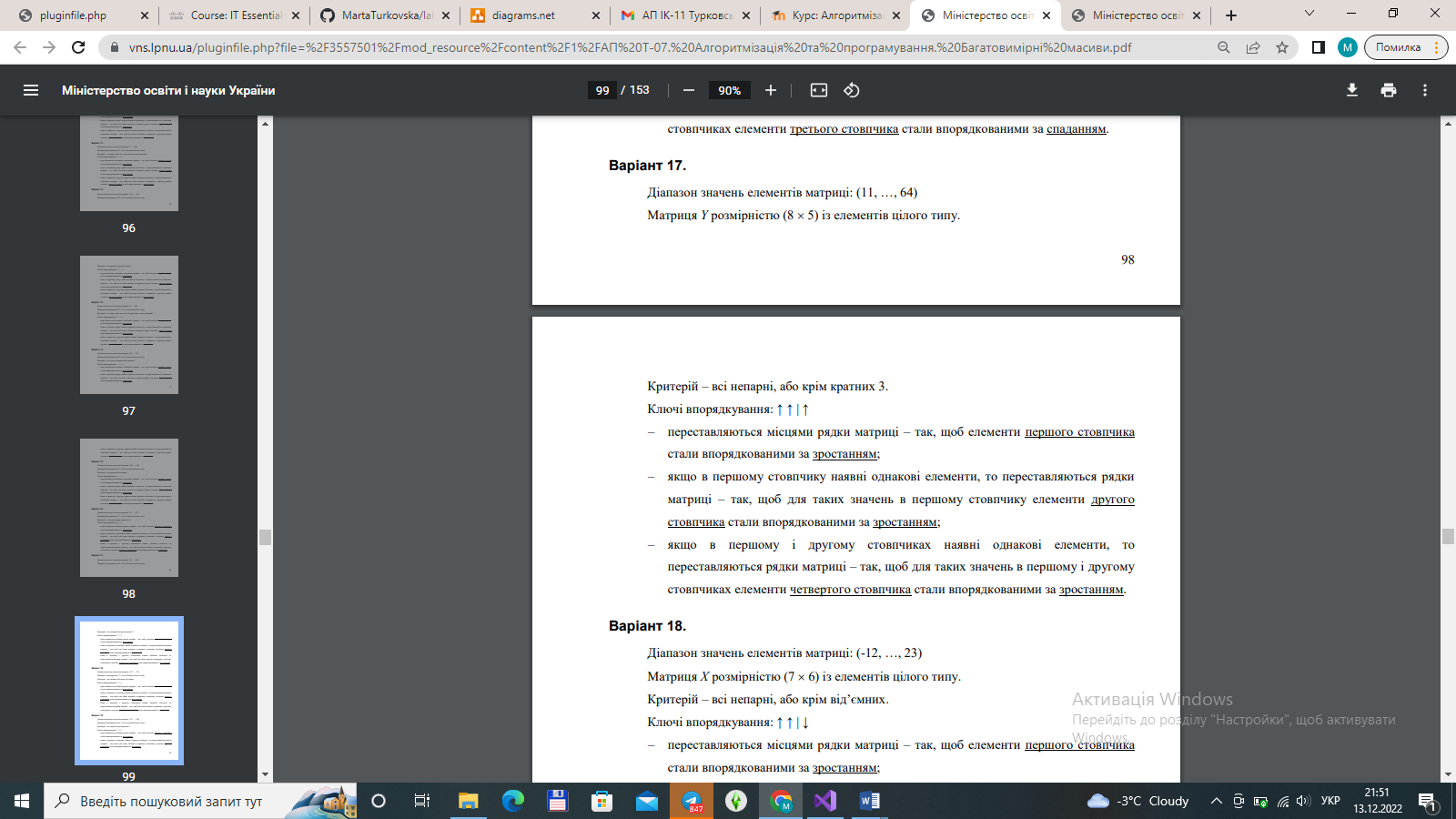
- (завдання 1) вивести результат – обчислену кількість і суму, та вивести на екран модифіковану матрицю у вигляді окремої таблиці, використовуючи виведення з тими самими специфікаціями формату.

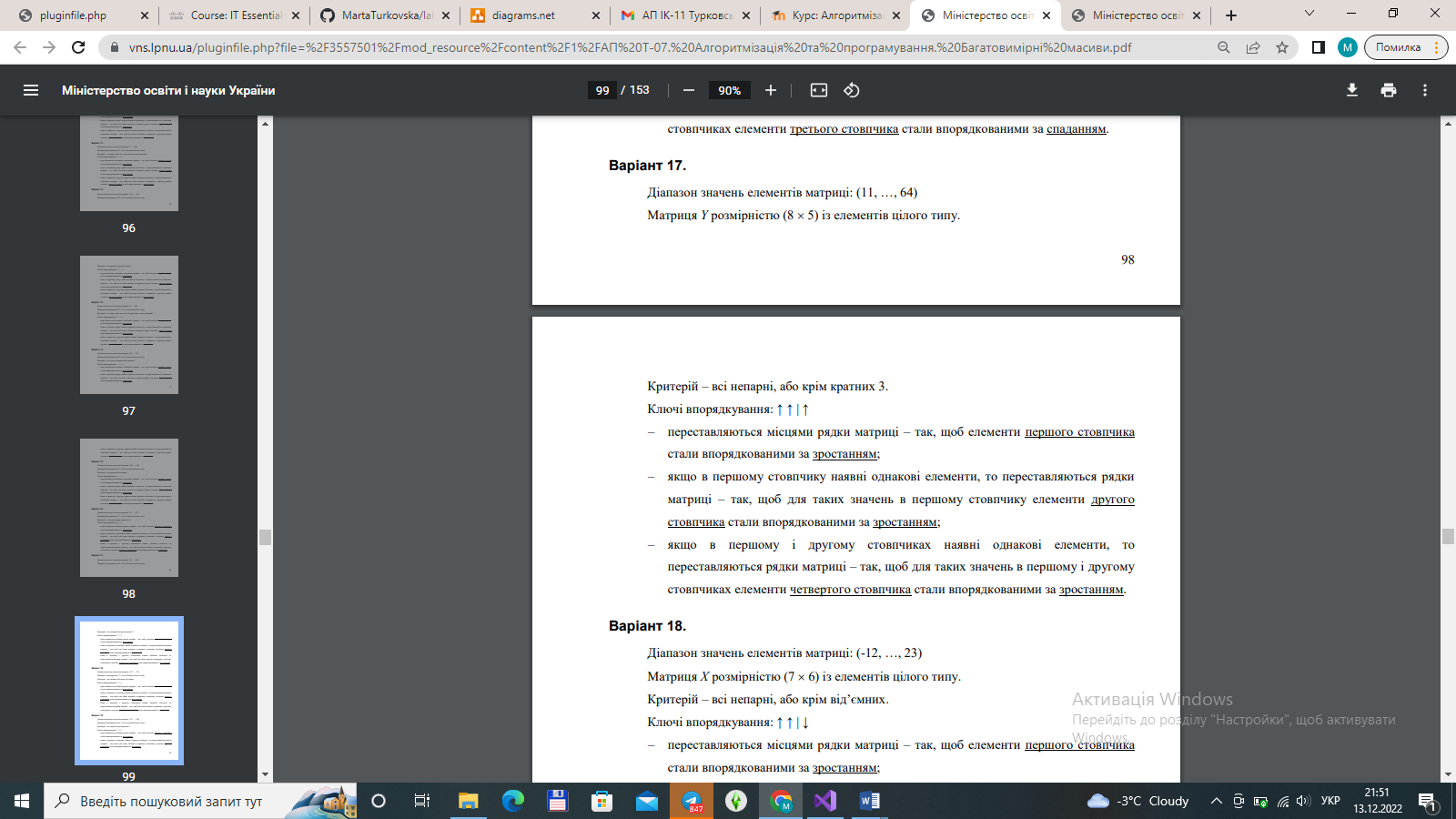
Усі вказані дії необхідно реалізувати за допомогою окремих функцій. Інформація у функції має передаватися лише за допомогою параметрів. Використання глобальних змінних – не допускається.

Кожна функція має виконувати лише одну роль, і ця роль має бути відображена у назві функції.

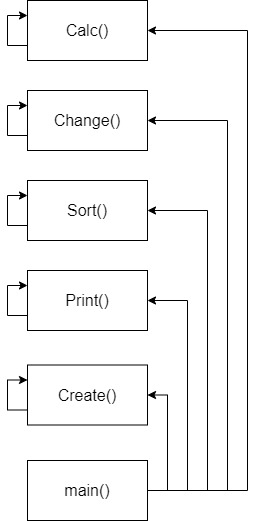
Рекурсивний та ітераційний способи – це 2 різні проекти, для яких потрібно 2 різних unit-тести і 2 різних звіти.

«Функція, яка повертає / обчислює / шукає ...» – має не виводити ці значення, а повернути їх у місце виклику як результат функції або як відповідний вихідний параметр.





**Структурна схема:**

****

**Текст програми:**

#include <iostream>

#include <iomanip>

#include <time.h>

using namespace std;

void Create(int\*\* y, const int rowCount, const int colCount, const int Low, const int High, int i, int j);

void Print(int\*\* y, const int rowCount, const int colCount, int i, int j);

void Sort(int\*\* y, const int rowCount, const int colCount, int i0, int i1);

void Change(int\*\* y, const int row1, const int row2, const int colCount, int j);

void Calc(int\*\* y, const int rowCount, const int colCount, int& S, int& k, int i, int j);

int main()

{

srand((unsigned)time(NULL));

int Low = 11;

int High = 64;

int rowCount = 8;

int colCount = 5;

int\*\* y = new int\* [rowCount];

for (int i = 0; i < rowCount; i++)

y[i] = new int[colCount];

Create(y, rowCount, colCount, Low, High, 0, 0);

Print(y, rowCount, colCount, 0, 0);

Sort(y, rowCount, colCount, 0, 0);

Print(y, rowCount, colCount, 0, 0);

int S = 0;

int k = 0;

Calc(y, rowCount, colCount, S, k, 0, 0);

cout << "S = " << S << endl;

cout << "k = " << k << endl;

Print(y, rowCount, colCount, 0, 0);

for (int i = 0; i < rowCount; i++)

delete[] y[i];

delete[] y;

return 0;

}

void Create(int\*\* y, const int rowCount, const int colCount, const int Low,

const int High, int i, int j)

{

if (i < rowCount) {

if (j < colCount) {

y[i][j] = Low + rand() % (High - Low + 1);

Create(y, rowCount, colCount, Low, High, i, j + 1);

}

Create(y, rowCount, colCount, Low, High, i + 1, j);

}

}

void Print(int\*\* y, const int rowCount, const int colCount, int i, int j)

{

cout << setw(4) << y[i][j];

if (j < colCount - 1)

Print(y, rowCount, colCount, i, j + 1);

else

if (i < rowCount - 1)

{

cout << endl;

Print(y, rowCount, colCount, i + 1, 0);

}

else

cout << endl << endl;

}

void Sort(int\*\* y, const int rowCount, const int colCount, int i0, int i1)

{

if (i0 < rowCount - 1) {

if (i1 < rowCount - i0 - 1) {

if ((y[0][i1] > y[0][i1 + 1])

||

(y[0][i1] == y[0][i1 + 1] &&

y[1][i1] > y[1][i1 + 1])

||

(y[0][i1] == y[0][i1 + 1] &&

y[1][i1] == y[1][i1 + 1] &&

y[3][i1] > y[3][i1 + 1]))

Change(y, i1, i1 + 1, rowCount, 0);

Sort(y, rowCount, colCount, i0, i1 + 1);

}

Sort(y, rowCount, colCount, i0 + 1, i1);

}

}

void Change(int\*\* y, const int row1, const int row2, const int colCount, int j)

{

int tmp;

if (j < colCount)

{

tmp = y[j][row1];

y[j][row1] = y[j][row2];

y[j][row2] = tmp;

Change(y, row1, row2, colCount, j + 1);

}

}

void Calc(int\*\* y, const int rowCount, const int colCount, int& S, int& k, int i, int j)

{

if (i < rowCount) {

if (j < colCount) {

if (y[i][j] % 2 != 0 && y[i][j] % 3 != 0)

{

S += y[i][j];

k++;

y[i][j] = 0;

Calc(y, rowCount, colCount, S, k, i, j + 1);

}

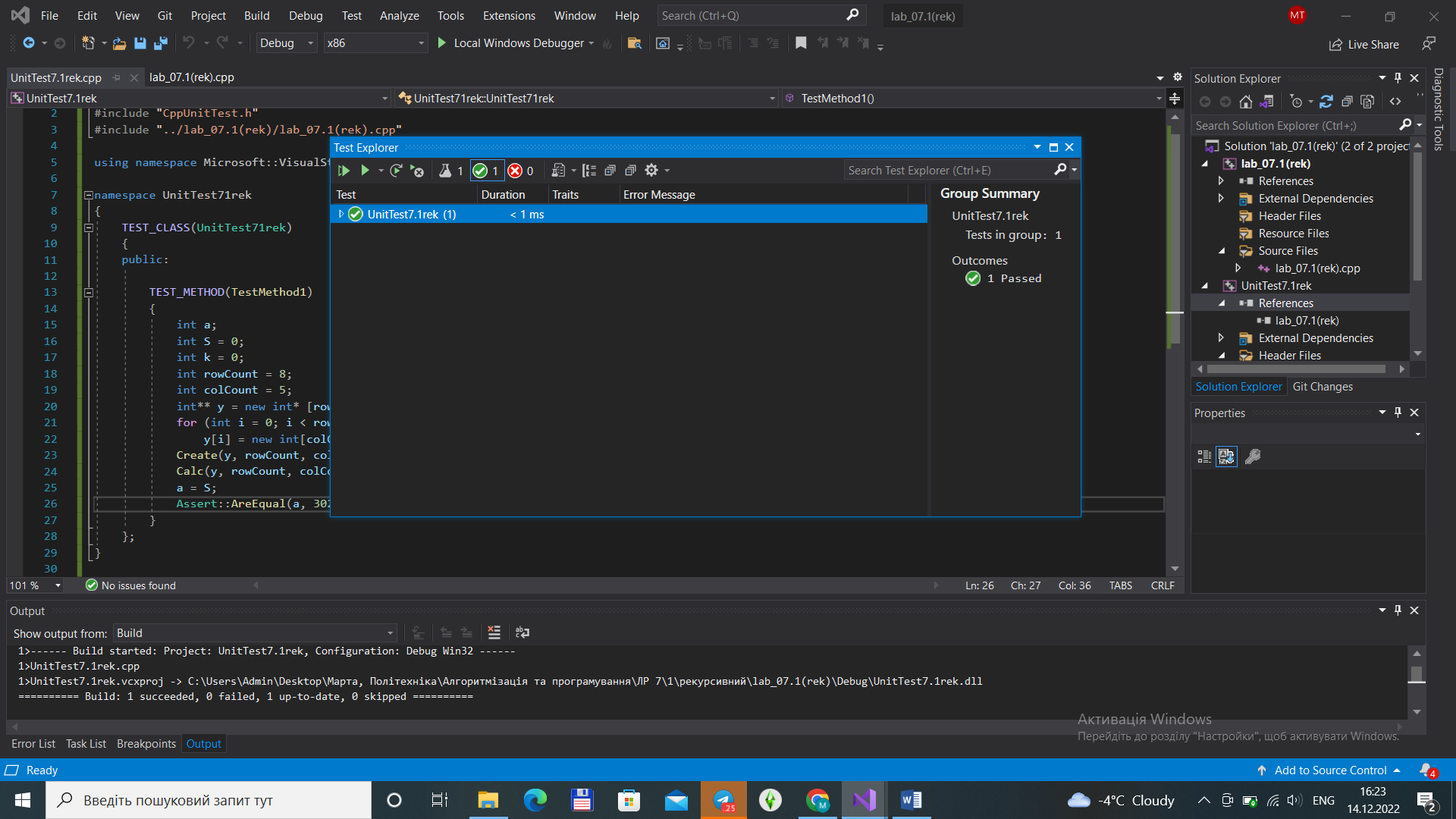
}

Calc(y, rowCount, colCount, S, k, i + 1, j);

}

}

**Результати unit-тесту:**



**Посилання на git-репозиторій:**

**Висновок:** навчилися обчислювати кількість, суму заданих елементів матриці; а також впорядковувати матрицю, переставляючи місцями її рядки / стовпчики.