МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний аерокосмічний університет
«Харківський авіаційний інститут»

Факультет систем управління літальних апаратів
Кафедра систем управління літальних апаратів

Лабораторна робота № 7

з дисципліни «Алгоритмізація та програмування»
на тему «Реалізація алгоритмів обробки двовимірних масивів мовою С ++»

ХАІ.301. 319а 18 ЛР

Виконав студент гр. _____319а_____

_____Тучак Владислав Олександрович_____

(підпис, дата) (П.І.Б.)

Перевірив

_____ к.т.н., доц. Олена ГАВРИЛЕНКО

(підпис, дата) (П.І.Б.)

2026

МЕТА РОБОТИ

Вивчити теоретичний матеріал з основ представлення двовимірних масивів (матриць) у мові С ++ і реалізувати декларацію, введення з консолі, обробку і

виведення в консоль матриць мовою С ++ в середовищі QtCreator.

ПОСТАНОВКА ЗАДАЧІ

Завдання 1. Вирішити завдання на аналіз і виведення елементів матриці.

Введення і виведення даних здійснити в командному вікні.Завдання Matrix28

Matrix28. Дана матриця розміру M × N. Знайти мінімальний серед максимальних елементів її стовпців.

Завдання 2. Перетворити матрицю відповідно до свого варіанту завдання Matrix71, розмір матриці і його елементи ввести з консолі. Вивести результати у консоль

Matrix71. Дана матриця розміру M × N. Продублювати стовпець матриці, що містить її мінімальний елемент.

Завдання 3. У функції main() організувати багаторазовий вибір одного з двох завдань. Кожне завдання описати окремою функцією без параметрів. Введення, виведення, обробку матриць реалізувати окремими функціями з параметрами.

ВИКОНАННЯ РОБОТИ

Завдання 1 (Таблиця 1) — Matrix28

Формулювання задачі. Дана матриця розміру M × N. Потрібно знайти мінімальний елемент серед максимальних елементів усіх стовпців матриці.

Вхідні дані.

Вводяться: кількість рядків `m` (тип int, допустимі значення 2…50), кількість стовпців `n` (тип int, допустимі значення 2…50), а також елементи матриці `a[m][n]` (тип int, цілі числа, вводяться користувачем з клавіатури).

Вихідні дані.

Виводиться введена матриця та значення `result` (тип int) — мінімальний серед максимумів стовпців.

Опис алгоритму.

Спочатку вводяться `m` та `n` з перевіркою, щоб розміри були в межах 2…50. Далі вводяться всі елементи матриці. Після цього програма по черзі розглядає кожний стовпець: для поточного стовпця знаходиться його максимальний елемент (переглядом усіх рядків). Коли максимуми всіх стовпців знайдені, з них обирається найменше значення — це і є шуканий результат. Наприкінці програма виводить матрицю та отриману відповідь.

Завдання 2 (Таблиця 2) — Matrix71

Формулювання задачі. Дана матриця розміру M × N. Потрібно продублювати стовпець матриці, який містить мінімальний елемент цієї матриці.

Вхідні дані.

Вводяться: кількість рядків `m` (тип int, допустимі значення 2…50), кількість стовпців `n` (тип int, допустимі значення 2…50), а також елементи матриці `a[m][n]` (тип int, цілі числа, вводяться користувачем). Для виконання дублювання потрібно, щоб було місце для додаткового стовпця, тому якщо `n = 50`, дублювання виконати неможливо.

Вихідні дані.

Виводиться початкова матриця, а також матриця після дублювання стовпця. Якщо `n = 50`, виводиться повідомлення про неможливість дублювання через обмеження максимальної кількості стовпців.

Опис алгоритму.

Після введення `m`, `n` та елементів матриці програма знаходить мінімальний

елемент у всій матриці, переглядаючи всі значення `a[i][j]`. Разом із мінімальним значенням запам’ятовується номер стовпця `minCol`, у якому цей мінімум знаходиться. Далі перевіряється, чи можна додати новий стовпець: якщо `n = 50`, програма повідомляє про помилку і завершує виконання завдання. Якщо `n < 50`, у кожному рядку виконується зсув елементів праворуч, щоб звільнити позицію `minCol + 1`, після чого в цю позицію копіюється значення з `minCol`. Таким чином, стовпець з мінімальним елементом дублюється (дубль вставляється одразу праворуч від оригіналу). Після цього програма виводить матрицю до та після перетворення.

ВИСНОВКИ

Було вивчено принципи роботи з двовимірними масивами (матрицями) в мові C++ та реалізовано введення і виведення матриці з перевіркою коректності розмірів. На практиці відпрацьовано алгоритми: пошук мінімального серед максимальних елементів стовпців (Matrix28) та дублювання стовпця, що містить мінімальний елемент матриці (Matrix71). У процесі виконання виникали труднощі з коректним зсувом елементів під час вставки нового стовпця, однак їх було усунуто шляхом поелементного зсуву справа наліво без використання допоміжної матриці.

ДОДАТОК А
Лістинг коду програми

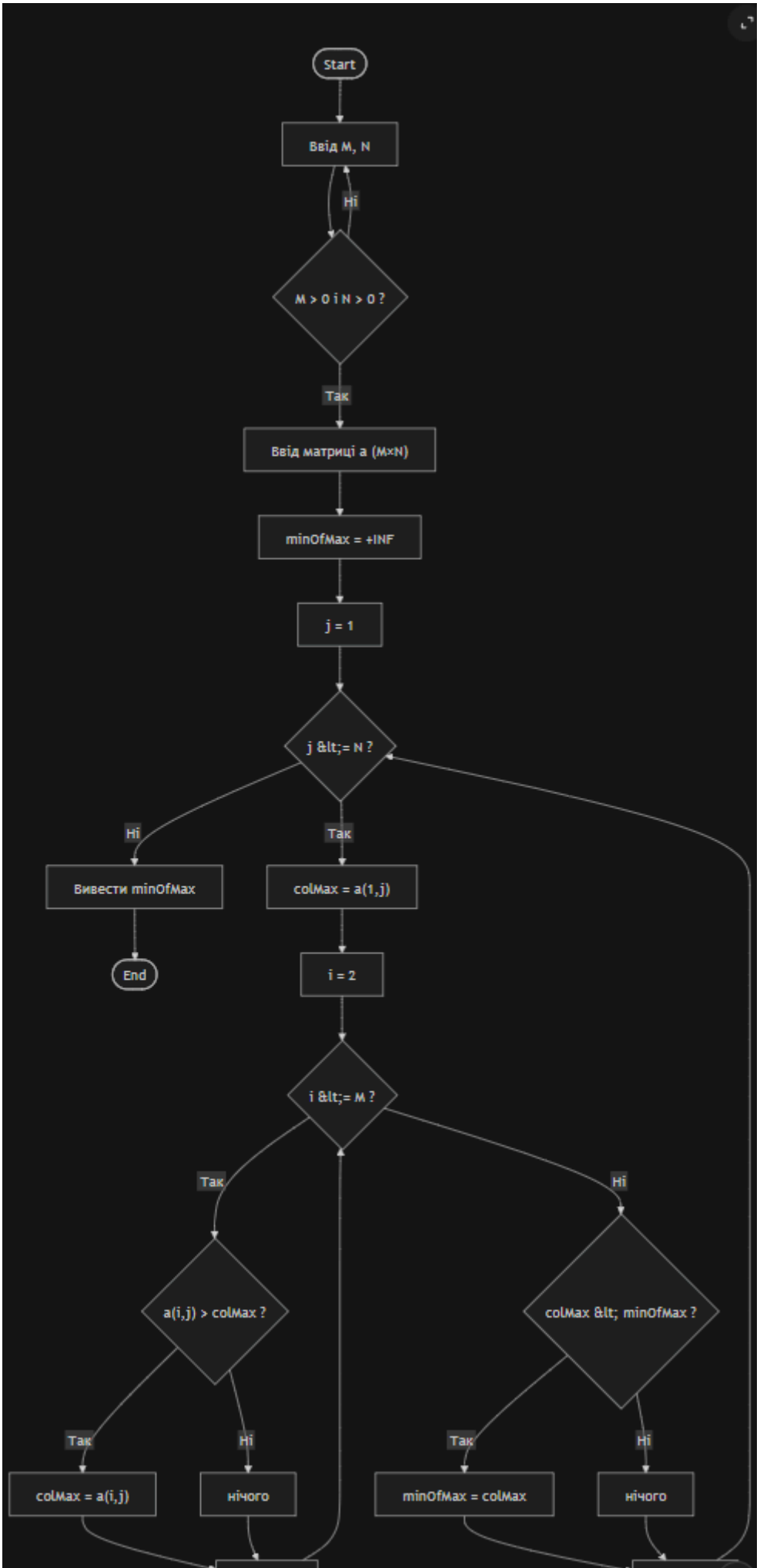
```
 #include <iostream>
#include <limits>
using namespace std;

const int MAX_M = 50;
const int MAX_N = 50;

// Input matrix (2..50) with validation
void inputMatrix(int a[MAX_M][MAX_N], int &m, int &n) {
    while (true) {
        cout << "Enter number of rows m (2..50): ";
        cin >> m;
        cout << "Enter number of columns n (2..50): ";
        cin >> n;

        if (!cin) {
            cin.clear();
            cin.ignore(numeric_limits<streamsize>::max(), '\n');
            cout << "Input error. Try again.\n\n";
            continue;
        }

        if (m < 2 || m > 50 || n < 2 || n > 50) {
            cout << "Error: m and n must be in range 2..50. Try again.\n\n";
            continue;
        }
        break;
    }

    cout << "Enter matrix elements (" << m << "x" << n << "):\n";
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            cin >> a[i][j];
        }
    }
}

// Print matrix
void printMatrix(const int a[MAX_M][MAX_N], int m, int n) {
    cout << "Matrix (" << m << "x" << n << "):\n";
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            cout << a[i][j] << "\t";
        }
```

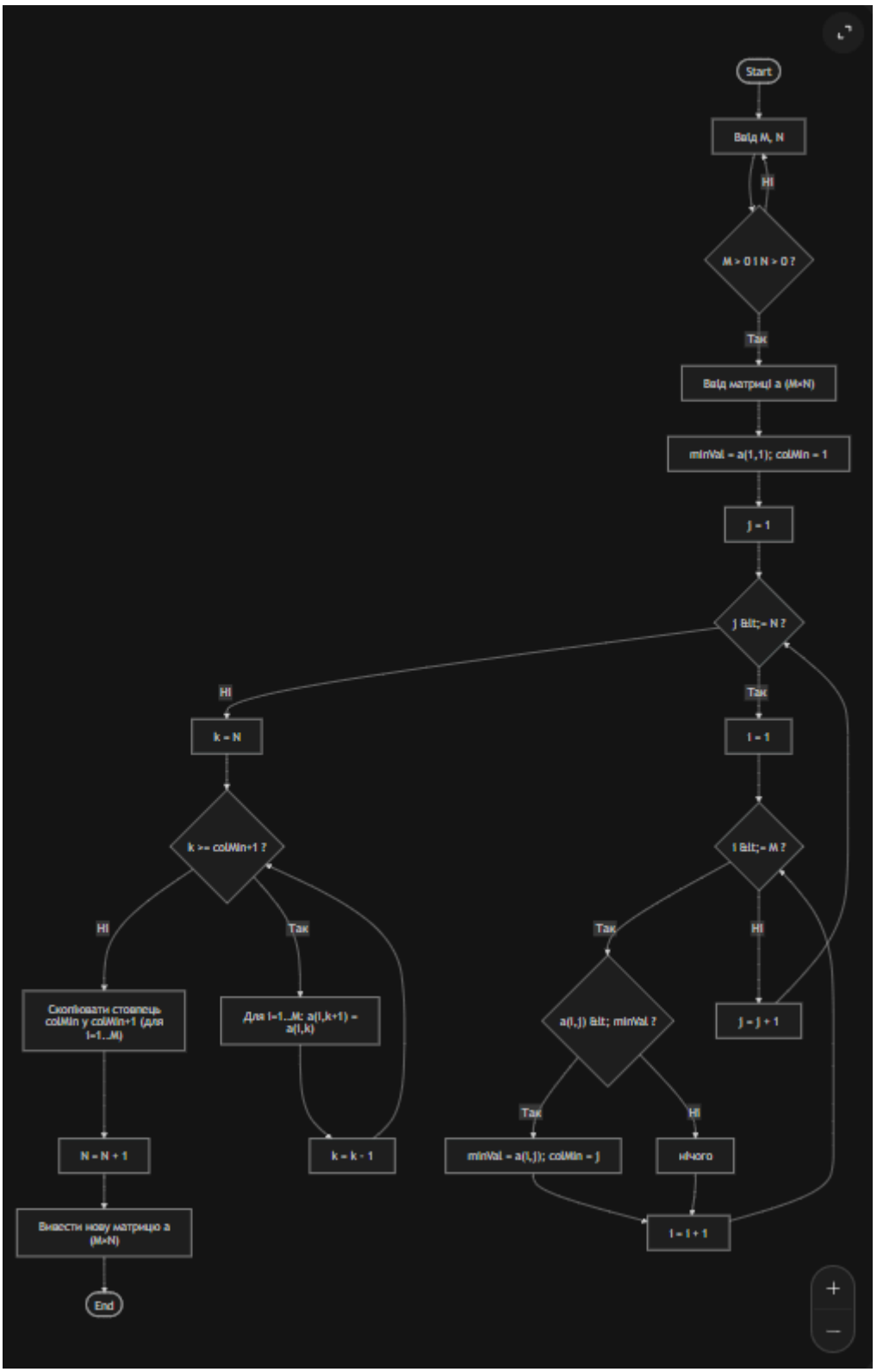
```
        cout << "\n";
    }
}

// ====================== Table 1: Matrix28 ======================
// Given MxN matrix. Find the minimum among the maximum elements of its columns.
int minOfColumnMax(const int a[MAX_M][MAX_N], int m, int n) {
    int answer = numeric_limits<int>::max();

    for (int j = 0; j < n; j++) {
        int colMax = a[0][j];
        for (int i = 1; i < m; i++) {
            if (a[i][j] > colMax) colMax = a[i][j];
        }
        if (colMax < answer) answer = colMax;
    }
    return answer;
}

void taskMatrix28() {
    int a[MAX_M][MAX_N];
    int m, n;

    cout << "\n=== Task Matrix28 (Table 1) ===\n";
    cout << "Find the minimum among the maximum elements of the columns.\n\n";

    inputMatrix(a, m, n);
    cout << "\n";
    printMatrix(a, m, n);

    int result = minOfColumnMax(a, m, n);
    cout << "\nMinimum among column maximums = " << result << "\n\n";
}

// ====================== Table 2: Matrix71 ======================
// Given MxN matrix. Duplicate the column that contains the minimal element of
the matrix.
// Implementation: duplicate column is inserted immediately to the RIGHT of that
column.
bool duplicateColumnWithGlobalMin(int a[MAX_M][MAX_N], int m, int &n) {
    if (n >= MAX_N) return false; // no space for new column

    int minVal = a[0][0];
    int minCol = 0;

    // find global minimum and its column
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            if (a[i][j] < minVal) {
                minVal = a[i][j];
```

```
                minCol = j;
            }
        }
    }

    // shift right and insert duplicate (row by row)
    for (int i = 0; i < m; i++) {
        for (int j = n; j > minCol + 1; j--) {
            a[i][j] = a[i][j - 1];
        }
        a[i][minCol + 1] = a[i][minCol];
    }

    n++;
    return true;
}

void taskMatrix71() {
    int a[MAX_M][MAX_N];
    int m, n;

    cout << "\n=== Task Matrix71 (Table 2) ===\n";
      cout << "Duplicate the column that contains the minimal element of the
matrix.\n";
       cout << "(The duplicate is inserted immediately to the RIGHT of that
column.)\n\n";

    inputMatrix(a, m, n);
    cout << "\nInitial ";
    printMatrix(a, m, n);

    bool ok = duplicateColumnWithGlobalMin(a, m, n);
    if (!ok) {
        cout << "\nCannot duplicate: n = 50 (no space for a new column).\n\n";
        return;
    }

    cout << "\nAfter duplication ";
    printMatrix(a, m, n);
    cout << "\n";
}

// ==================== Menu ====================
int main() {
    int choice;

    do {
        cout << "============================\n";
        cout << "Lab work: Matrices\n";
        cout << "1 - Matrix28 (Table 1)\n";
```

```
        cout << "2 - Matrix71 (Table 2)\n";
        cout << "0 - Exit\n";
        cout << "Your choice: ";
        cin >> choice;

        if (!cin) {
            cin.clear();
            cin.ignore(numeric_limits<streamsize>::max(), '\n');
            choice = -1;
        }

        switch (choice) {
            case 1: taskMatrix28(); break;
            case 2: taskMatrix71(); break;
            case 0: cout << "Exit.\n"; break;
            default: cout << "Invalid choice. Try again.\n\n"; break;
        }
    } while (choice != 0);

    return 0;
}
```

Діаграми


Start
Ввід M, N
Ні
M > 0 і N > 0 ?
Так
Ввід матриці a (M×N)
minOfMax = +INF
j = 1
j <= N ?
Ні
Так
Вивести minOfMax
colMax = a(1,j)
End
i = 2
i <= M ?
Так
Ні
a(i,j) > colMax ?
colMax < minOfMax ?
Так
Ні
Так
Ні
colMax = a(i,j)
нічого
minOfMax = colMax
нічого


Start
Ввід M, N
HI
M > 0 I N > 0 ?
Так
Ввід матриці a (M×N)
minVal = a(1,1); colMin = 1
j = 1
j <= N ?
HI
Так
k = N
i = 1
k >= colMin+1 ?
i <= M ?
HI
Так
Так
HI
Скопіювати стовпець colMin у colMin+1 (для i=1..M)
Для i=1..M: a(i,k+1) = a(i,k)
a(i,j) < minVal ?
j = j + 1
Так
HI
N = N + 1
k = k - 1
minVal = a(i,j); colMin = j
нічого
Вивести нову матрицю a (M×N)
i = i + 1
End

ДОДАТОК Б
Скрін-шоти вікна виконання програми

Link to this code: [copy]

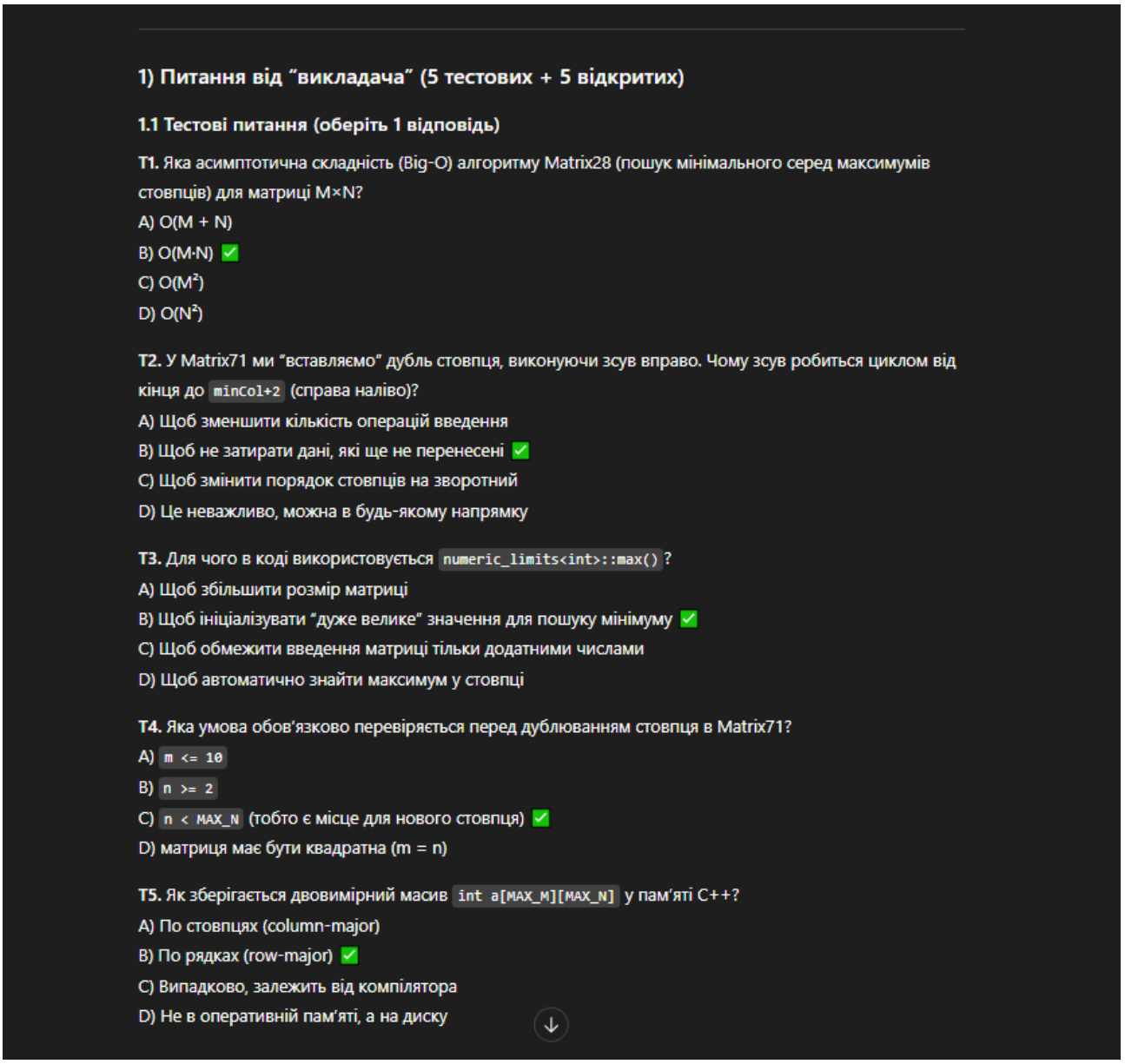
options | compilation | execution

```
============================
Lab work: Matrices
1 - Matrix28 (Table 1)
2 - Matrix71 (Table 2)
0 - Exit
Your choice:
```

```
Your choice: 1

=== Task Matrix28 (Table 1) ===
Find the minimum among the maximum elements of the columns.

Enter number of rows m (2..50): 4
Enter number of columns n (2..50): 5
Enter matrix elements (4x5):
6
234
6
4

6
5
4
6
4
3
6
3
5




4365

3246
346
34
643
62346
236

Matrix (4x5):
6       234     6       4       6
5       4       6       4       3
6       3       5       4365    3246
346     34      643     62346   236

Minimum among column maximums = 234

============================
Lab work: Matrices
1 - Matrix28 (Table 1)
2 - Matrix71 (Table 2)
0 - Exit
Your choice: 3246
Invalid choice. Try again.

============================
Lab work: Matrices
1 - Matrix28 (Table 1)
2 - Matrix71 (Table 2)
0 - Exit
```

```
============================
Lab work: Matrices
1 - Matrix28 (Table 1)
2 - Matrix71 (Table 2)
0 - Exit
Your choice: 2

=== Task Matrix71 (Table 2) ===
Duplicate the column that contains the minimal element of the matrix.
(The duplicate is inserted immediately to the RIGHT of that column.)

Enter number of rows m (2..50): 2
Enter number of columns n (2..50): 2
Enter matrix elements (2x2):
4
4
4
4

Initial Matrix (2x2):
4       4
4       4

After duplication Matrix (2x3):
4       4       4
4       4       4

----------------------------
```

ДОДАТОК В
Діалог з ШІ для самоаналізу



1) Питання від "викладача" (5 тестових + 5 відкритих)

1.1 Тестові питання (оберіть 1 відповідь)

T1. Яка асимптотична складність (Big-O) алгоритму Matrix28 (пошук мінімального серед максимумів стовпців) для матриці M×N?

A) O(M + N)

B) O(M·N) ✅

C) O(M^2)

D) O(N^2)

T2. У Matrix71 ми "вставляємо" дубль стовпця, виконуючи зсув вправо. Чому зсув робиться циклом від кінця до `minCol+2` (справа наліво)?

A) Щоб зменшити кількість операцій введення

B) Щоб не затирати дані, які ще не перенесені ✅

C) Щоб змінити порядок стовпців на зворотний

D) Це неважливо, можна в будь-якому напрямку

T3. Для чого в коді використовується `numeric_limits<int>::max()`?

A) Щоб збільшити розмір матриці

B) Щоб ініціалізувати "дуже велике" значення для пошуку мінімуму ✅

C) Щоб обмежити введення матриці тільки додатними числами

D) Щоб автоматично знайти максимум у стовпці

T4. Яка умова обов'язково перевіряється перед дублюванням стовпця в Matrix71?

A) `m <= 10`

B) `n >= 2`

C) `n < MAX_N` (тобто є місце для нового стовпця) ✅

D) матриця має бути квадратна (m = n)

T5. Як зберігається двовимірний масив `int a[MAX_M][MAX_N]` у пам'яті C++?

A) По стовпцях (column-major)

B) По рядках (row-major) ✅

C) Випадково, залежить від компілятора

D) Не в оперативній пам'яті, а на диску

1.2 Відкриті питання (пояснити словами)

B1. Поясніть ідею задачі Matrix28. Чому потрібно спочатку знайти максимуми стовпців, а потім взяти мінімум серед них? Наведіть короткий приклад.

B2. Опишіть, як у Matrix71 знаходиться стовпець `minCol`. Що робить програма, якщо мінімальних елементів кілька і вони в різних стовпцях?

B3. Поясніть, чому в Matrix71 заборонено дублювання, якщо `n = 50`. Що станеться без цієї перевірки?

B4. Які перевірки введення реалізовані у програмі? Поясніть, для чого робиться `cin.clear()` та `cin.ignore(...)`.

B5. Як би ви модифікували Matrix71, щоб дубльований стовпець вставлявся НЕ праворуч, а в кінець матриці? Опишіть зміни в логіці.

---

2) Аналіз відповідей, оцінювання та ймовірність використання ШІ

Нижче — **готовий приклад "відповідей студента"** (можеш лишити або трохи перефразувати під себе) + оцінка.

Тестові (відповіді студента)

- **T1:** B) O(M·N)
- **T2:** B) Щоб не затирати дані, які ще не перенесені
- **T3:** B) Для ініціалізації великого значення при пошуку мінімуму
- **T4:** C) `n < MAX_N`
- **T5:** B) По рядках (row-major)

Оцінювання тестових (5/5 кожне, якщо правильно):
T1: 5, T2: 5, T3: 5, T4: 5, T5: 5
Ймовірність ШІ: низька/середня (короткі факти), зниження не застосовується.