PROIECT DE PROGRAME NR. 4. LUCRUL CU TABELE BIDIMENSIONALE DE NUMERE

I. Formularea problemei

De efectuat următoarele operaţii conform variantei:

1. De determinat într-o matrice valorile minimă şi maximă şi poziţiile lor.

2. De interschimbat 2 linii (variantele impare) sau 2 coloane (variantele pare).

3. De adăugat o linie și / sau coloană, de completat cu caracteristica conform variantei.

4. De aranjat o matrice după linia sau coloana adăugată.

5. De generat o matrice conform variantei.

II. Indicaţii metodice

1. De scris funcţii pentru citirea, afişarea şi prelucrarea tabelelor statice, generate aleator sau citite de la tastatură.

2. Programul trebuie să fie universal, adică trebuie să se obţină rezultatul pentru orice date iniţiale.

3. Matricea poate fi de formă pătratică sau dreptunghiulară.

III. Programul

```
#include <iostream>
#include <cstdlib>
#include <conio.h>
#include <cmath>
#include <iomanip>

using namespace std;

int rrand(int range_min, int range_max) {
    return rand() % (range_max - range_min + 1) + range_min;
}

float **ArrayGenerator(size_t dim1, size_t dim2) {
    float **array = new float *[dim1];
    for (int i = 0; i < dim1; i++) {
        array[i] = new float[dim2];
    }
    return array;
}

void ArrayDestroyer(float **array, size_t dim1) {
    for (int i = 0; i < dim1; i++) {
        delete[] array[i];
    }
    delete[] array;
}

void Initialize(float **array, size_t dim1, size_t dim2) {
    for (size_t i = 0; i < dim1; ++i) {
        for (size_t j = 0; j < dim2; ++j) {
```

```
            array[i][j] = rrand(2, 10);
        }
    }
}

void Print(float **array, size_t dim1, size_t dim2) {
    for (size_t i = 0; i < dim1; ++i) {
        for (size_t j = 0; j < dim2; ++j) {
            cout << setw(8) << array[i][j];
        }
        cout << endl;
    }
}

void Position(float **array, size_t dim1, size_t dim2, float min, float max) {
    for (size_t i = 0; i < dim1; ++i) {
        for (size_t j = 0; j < dim2; ++j) {
            if (array[i][j] == max) {
                cout << "Valoareama maxima randul " << i + 1 << " coloana " << j +
1 << "\n";
            }
        }
    }
    cout << "\n";
    for (size_t i = 0; i < dim1; ++i) {
        for (size_t j = 0; j < dim2; ++j) {
            if (array[i][j] == min) {
                cout << "Valoareama minima randul " << i + 1 << " coloana " << j +
1 << "\n";
            }
        }
    }

    cout << "\n" << "Valoarea minima: " << min << "\n";
    cout << "Valoarea maxima: " << max << "\n";
}

void MinMax(float **array, size_t dim1, size_t dim2) {
    float min = array[0][0];
    float max = array[0][0];

    for (size_t i = 0; i < dim1; ++i) {
        for (size_t j = 0; j < dim2; ++j) {
            if (array[i][j] < min) {
                min = array[i][j];
            } else if (array[i][j] > max) {
                max = array[i][j];
            }
        }
    }
    Position(array, dim1, dim2, min, max);
}

void SwapLine(float **array, size_t dim1, size_t dim2, int first, int second) {
    float temp;
    for (size_t i = 0; i < dim1; ++i) {
        for (size_t j = 0; j < dim2; ++j) {
            if (i == first) {
                temp = array[second][j];
                array[second][j] = array[i][j];
                array[i][j] = temp;
            }
        }
```

```
    }
}

void AddColumn(float **array, size_t dim1, size_t &dim2) {
    dim2++;
    for (size_t i = 0; i < dim1; ++i) {
        float media = 0;
        for (size_t j = 0; j < dim2; ++j) {
            if (j < dim2 - 1) {
                media += array[i][j];
            }
            if (j == dim2 - 1) {
                array[i][j] = media / (dim2 - 1);
            }
        }

    }

}

void Sort(float **array, size_t dim1, size_t &dim2) {
    for (int i = 0; i < dim1 - 1; i++) {
        for (int j = 0; j < dim1 - i - 1; j++) {
            if (array[j][dim2 - 1] < array[j + 1][dim2 - 1]) {
                SwapLine(array, dim1, dim2, j, j + 1);
            }
        }
    }
}

void GenerateArray() {
    const int row = 3;
    const int column = 3;

    int array[row][column];

    for (int i = 0; i < row; i++) {
        for (int j = 0; j < column; j++) {
            array[i][j] = i + 1;
        }
    }

    for (int i = 0; i < row; i++) {
        for (int j = 0; j < column; j++) {
            cout << setw(4) << array[i][j];
        }
        cout << endl;
    }
}

int main() {
    size_t DIM1, DIM2; int first, second;
    cout << "Intruduceti nr de examane: " << "\n";
    cin >> DIM1;
    cout << "Intruduceti nr de studenti: " << "\n";
    cin >> DIM2;

    float **matrix;
    matrix = ArrayGenerator(DIM1, DIM2);
    Initialize(matrix, DIM1, DIM2);

    cout<< "\n" << "Tabelul generat: " << "\n";
    Print(matrix, DIM1, DIM2);
```

```
    cout << "\n"<< "Valorile minime maxime: "<< "\n";
    MinMax(matrix, DIM1, DIM2);
    cout << "\n"<< "Intoduceti liniile ce trebuie intershimbate: "<< "\n";
    cin >> first;
    cin >> second;
    cout<< "\n" << "Tabelul obtinut dupa interschimbare: " << "\n";
    SwapLine(matrix, DIM1, DIM2, first - 1, second - 1);
    Print(matrix, DIM1, DIM2);
    cout<< "\n" << "Tabelul obtinut dupa adaugarea mediei: " << "\n";
    AddColumn(matrix, DIM1, DIM2);
    Print(matrix, DIM1, DIM2);
    cout<< "\n" << "Tabelul obtinut dupa sortarea dupa medie: " << "\n";
    Sort(matrix, DIM1, DIM2);
    Print(matrix, DIM1, DIM2);
    cout<< "\n" << "Tabelul generat: " << "\n";
    GenerateArray();
    ArrayDestroyer(matrix, DIM1);
    return 0;
}
```

IV. Rezultate

Intruduceti nr de examane:

6

Intruduceti nr de studenti:

5

Tabelul generat:

7 10 9 6 10

3 5 2 9 4

10 4 9 8 9

7 9 10 5 2

2 8 7 2 6

9 8 7 10 7

Valorile minime maxime:

Valoareama maxima randul 1 coloana 2

Valoareama maxima randul 1 coloana 5

Valoareama maxima randul 3 coloana 1

Valoareama maxima randul 4 coloana 3

Valoareama maxima randul 6 coloana 4

Valoareama minima randul 2 coloana 3

Valoareama minima randul 4 coloana 5

Valoareama minima randul 5 coloana 1

Valoareama minima randul 5 coloana 4

Valoarea minima: 2

Valoarea maxima: 10

Intoduceti liniile ce trebuie intershimbate:

1

2

Tabelul obtinut dupa interschimbare:

3	5	2	9	4
7	10	9	6	10
10	4	9	8	9
7	9	10	5	2
2	8	7	2	6
9	8	7	10	7

Tabelul obtinut dupa adaugarea mediei:

3	5	2	9	4	4.6
7	10	9	6	10	8.4
10	4	9	8	9	8
7	9	10	5	2	6.6
2	8	7	2	6	5
9	8	7	10	7	8.2

Tabelul obtinut dupa sortarea dupa medie:

7	10	9	6	10	8.4
9	8	7	10	7	8.2
10	4	9	8	9	8
7	9	10	5	2	6.6
2	8	7	2	6	5
3	5	2	9	4	4.6

Tabelul generat:

1	1	1
2	2	2
3	3	3