**Додаткові завдання:**

1. Получить матрицу:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 0 | 0 | … | 0 |
| 0 | 3 | 0 | … | 0 |
| 0 | 0 | 5 | … | 0 |
| … | … | … | … |  |
| 0 | 0 | 0 | … | 21 |

1. Получить матрицу:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | … | 10 |
| 0 | 11 | 12 | … | 19 |
| 0 | 0 | 20 | … | 27 |
| … | … | … | … | … |
| 0 | 0 | 0 | … | 55 |

1. Составить алгоритм заполнения квадратной матрицы порядка n натуральными числами 1, 2, 3, …, n2 «по спирали» по часовой стрелке, начиная с центрального элемента.

Лістинг коду

#include "pch.h"

#include <iostream>

#include <iomanip>

using namespace std;

int main()

{

    int Solution;

    cout << "Choose question from 1 to 3: ";

    cin >> Solution;

    cout << '\n';

    switch (Solution) {

    case 1: {

        int k = 1, Matrix[11][11]{ 0 };

        for (int i = 0; i < 11; i++, cout << "\n\n")

            for (int j = 0; j < 11; j++) {

                if (i == j) { cout << setw(5) << k, k += 2; }

                else cout << setw(5) << Matrix[i][j];

            }

        break;

    }

    case 2: {

        int k = 0;

        for (int i = 0; i < 10; i++, cout << "\n\n")

            for (int j = 0; j < 10; j++) {

                if (i > j) cout << setw(5) << "0";

                else cout << setw(5) << ++k;

            }

        break;

    }

    case 3: {

        int Line;

        do {

            cout << "Choose colution <one or two>: ";

            cin >> Line;

        } while (Line != 1 && Line != 2);

        int \_SquareValuer;

        cout << "Enter n (odd) for square matrix: ";

        do {

            cin >> \_SquareValuer;

            if (\_SquareValuer % 2 == 0) cout << "Please check condition: ";

        } while (\_SquareValuer % 2 == 0);

        #define SqVal \_SquareValuer

        int  \_InternalCounter = \_SquareValuer;

        switch (Line) {

        case 1: {

            int CycleStep = SqVal \* SqVal;

            int \*\*\_SquareMatrix = new int \*[SqVal];

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

                \_SquareMatrix[i] = new int[SqVal];

            for (int i = -  1, j = 0; CycleStep > 0;) {

                for (int \_OutIC = 0; \_OutIC < \_InternalCounter; CycleStep--, \_OutIC++) {

                    i++;

                    \_SquareMatrix[i][j] = CycleStep;

                }

                \_InternalCounter--;

                for (int \_OutIC = 0; \_OutIC < \_InternalCounter; CycleStep--, \_OutIC++) {

                    j++;

                    \_SquareMatrix[i][j] = CycleStep;

                }

                for (int \_OutIC = 0; \_OutIC < \_InternalCounter; CycleStep--, \_OutIC++){

                    i--;

                    \_SquareMatrix[i][j] = CycleStep;

                }

                \_InternalCounter--;

                for (int \_OutIC = 0; \_OutIC < \_InternalCounter; CycleStep--, \_OutIC++) {

                    j--;

                    \_SquareMatrix[i][j] = CycleStep;

                }

            }

            cout << '\n';

            for (int i = 0; i < SqVal; i++, cout << "\n\n")

                for (int j = 0; j < SqVal; j++)

                    cout << setw(5) << \_SquareMatrix[i][j];

            break;

        }

        case 2: {

            int \*\*\_SquareMatrix = new int \*[SqVal];

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

                    \_SquareMatrix[i] = new int[SqVal];

                int CycleStep = SqVal \* SqVal;

                int \_Step, \_StepInc = 0;

                int \_Grad = 1, \_GradInc = 1;

                for (int i = -1, j = 0; CycleStep > 0;){

                    \_Step = \_InternalCounter - \_StepInc;

                    if (\_Step % 2 == 0) \_Grad = 2 + \_GradInc;

                    else \_Grad = 4 - 3 \* \_GradInc;

                    for (int EmbInc = 0; EmbInc < \_Step; CycleStep--, EmbInc++) {

                        switch (\_Grad) {

                        case 1: { i++; break; }

                        case 2: { j++; break; }

                        case 3: { i--; break; }

                        case 4: { j--; break; }

                        } \_SquareMatrix[i][j] = CycleStep;

                    }

                    if (\_Grad % 2 != 0) { \_GradInc = 0, \_StepInc++; }

                    else \_GradInc++;

                }

                cout << '\n';

                for (int i = 0; i < SqVal; i++, cout << "\n\n")

                    for (int j = 0; j < SqVal; j++)

                        cout << setw(5) << \_SquareMatrix[i][j];

                break;

            }

        }

        break;

    }

    default:

        cout << "Incorrect value!\n";

        break;

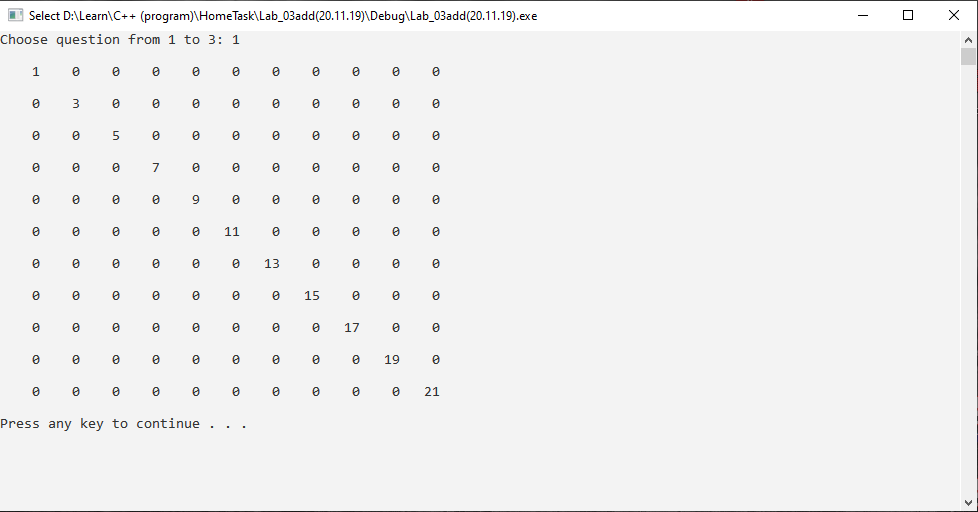
    }

    system("pause");

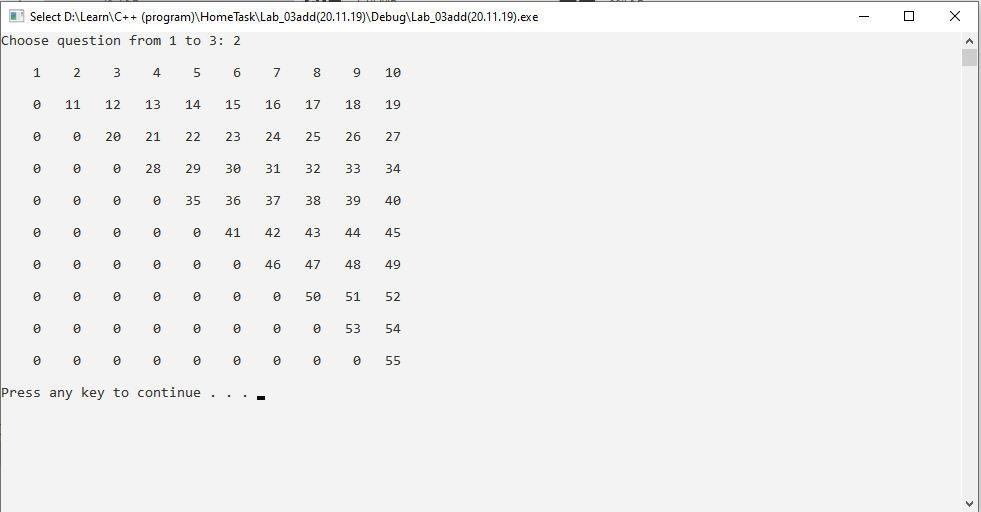
    return 0;

}

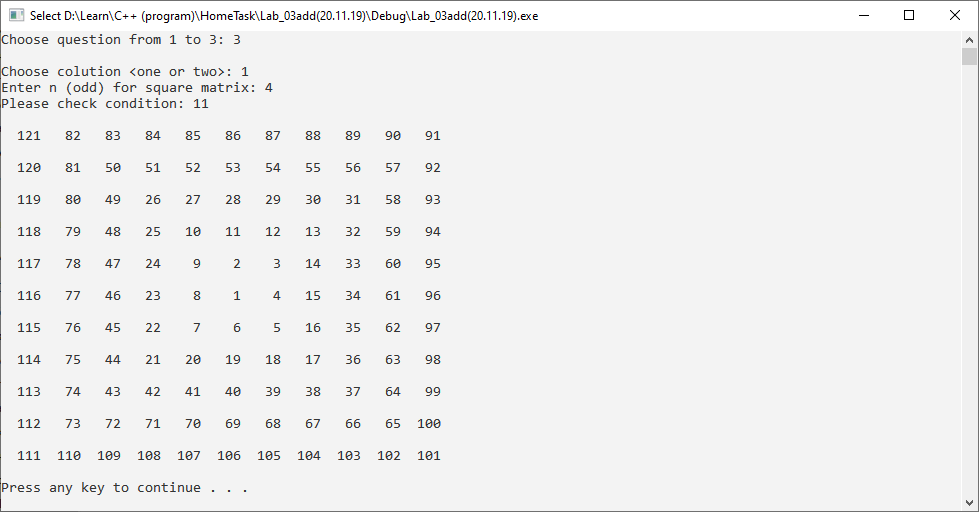
Лістинг виконання програми (завдання 1)



Лістинг виконання програми (завдання 2)



Лістинг виконання програми (завдання 3.1)



Лістинг виконання програми (завдання 3.2)

