**Архітектура обчислювальних систем та схемотехніка**

***Лабораторна робота № 11***

Виконала:

Cтудентка групи ПМО-11

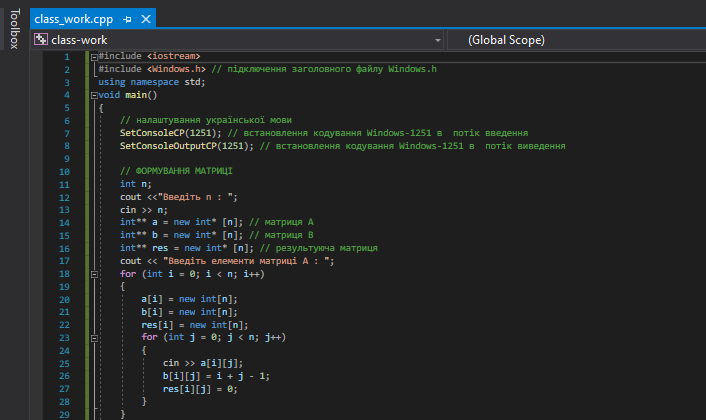
Кравець Ольга

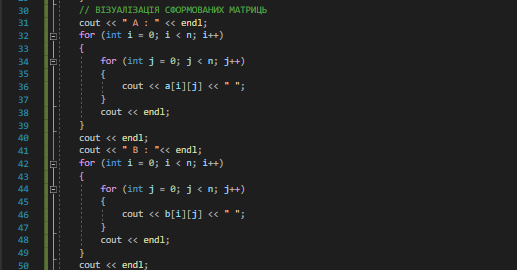
**Тема:** Масиви з індексуванням, стек, робота з матрицями .

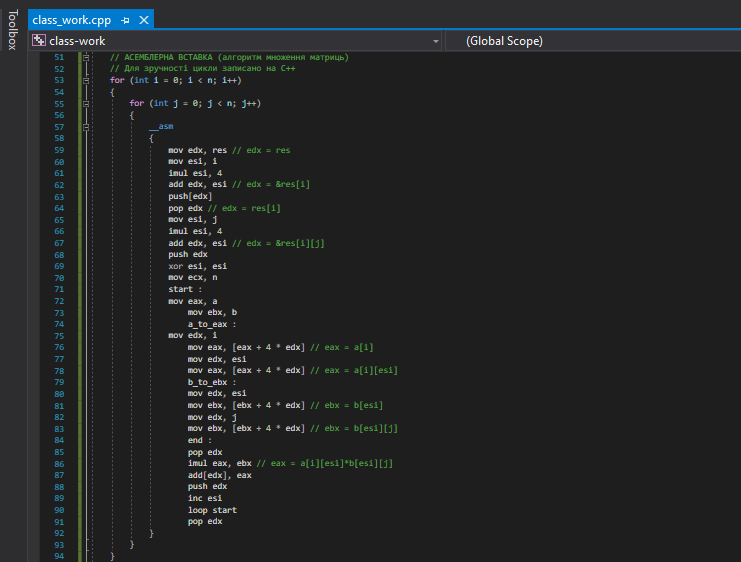
**Мета:** Використовуючи середовище Visual Studio, написати програму опрацювання масивів даних на мові Assembler.

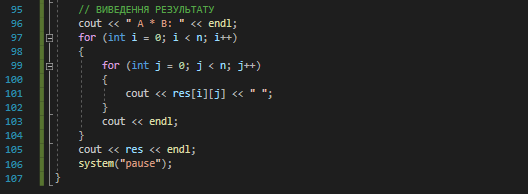
**Результат роботи:**

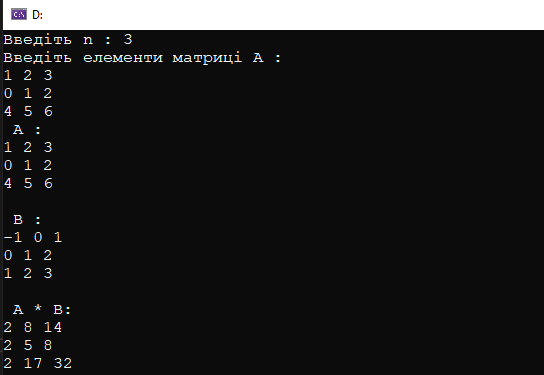
З використанням асемблерної вставки реалізувала програму для обчислення добутку матриць АВ, де А — задана матриця розмірності n x n, а елементи матриці В задаються формулою bij = i + j – 1:





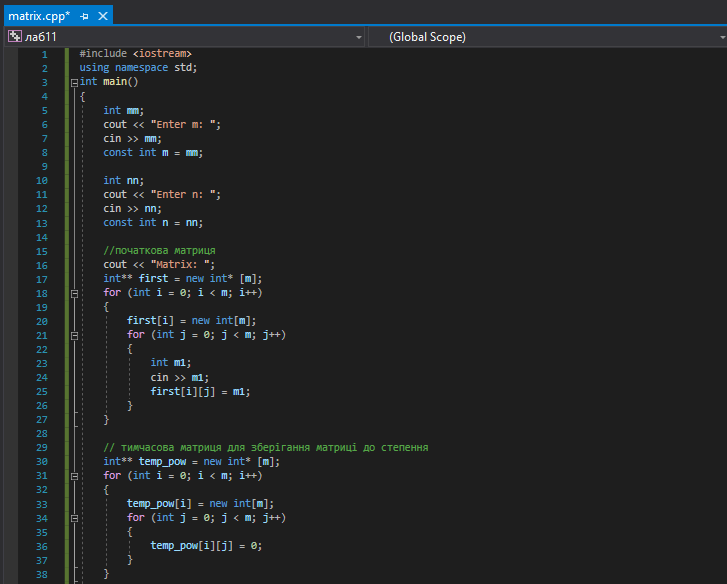


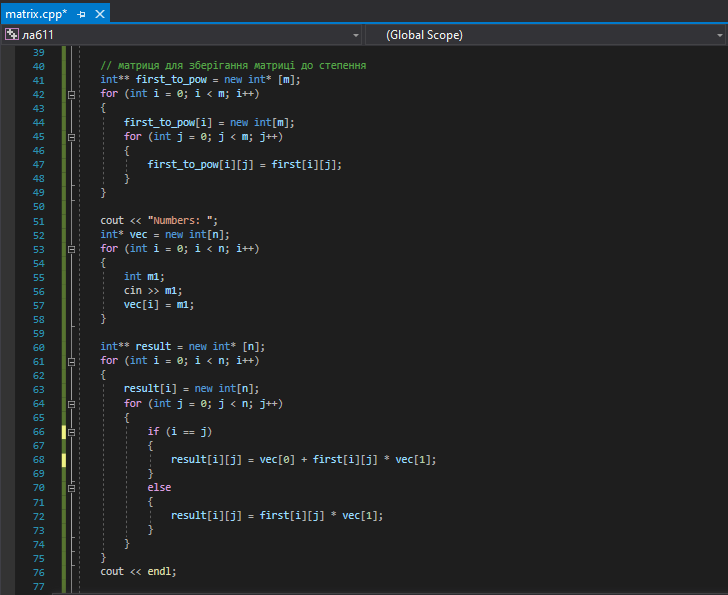


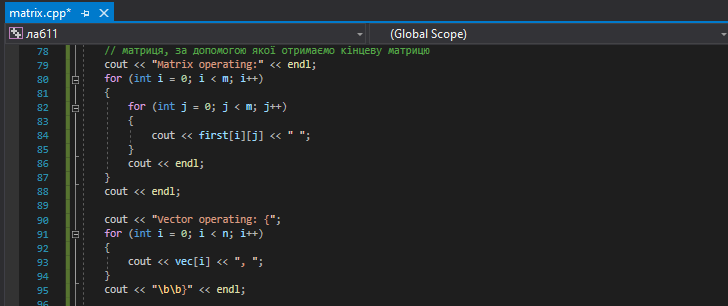


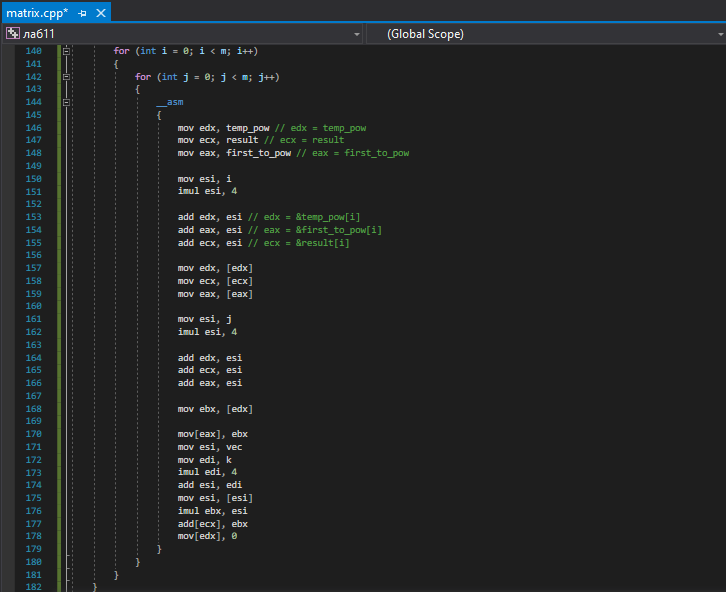
**Програма до завдання 10 варіанту:**

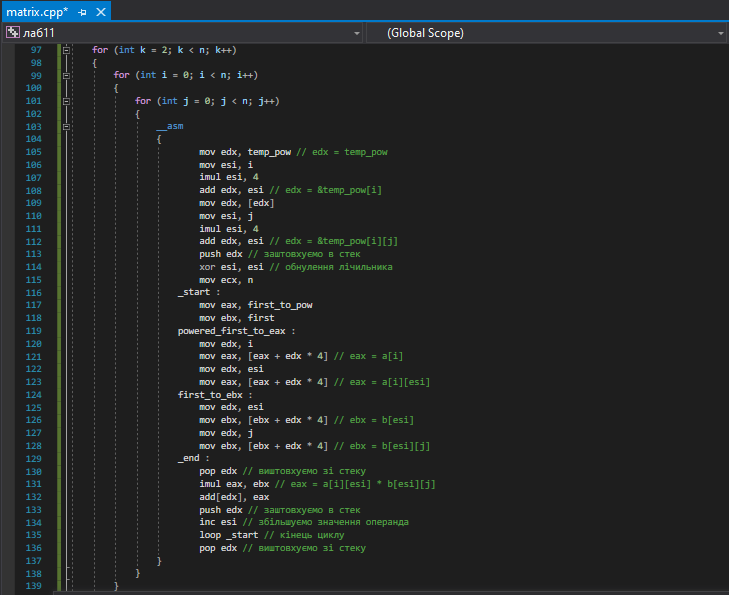
Задані квадратна матриця A порядку m, натуральне число n, дійсні числа pn, pn-1, ..., p0. Отримати матрицю pnAn + pn-1An-1 + ... + p1A1 + p0E.

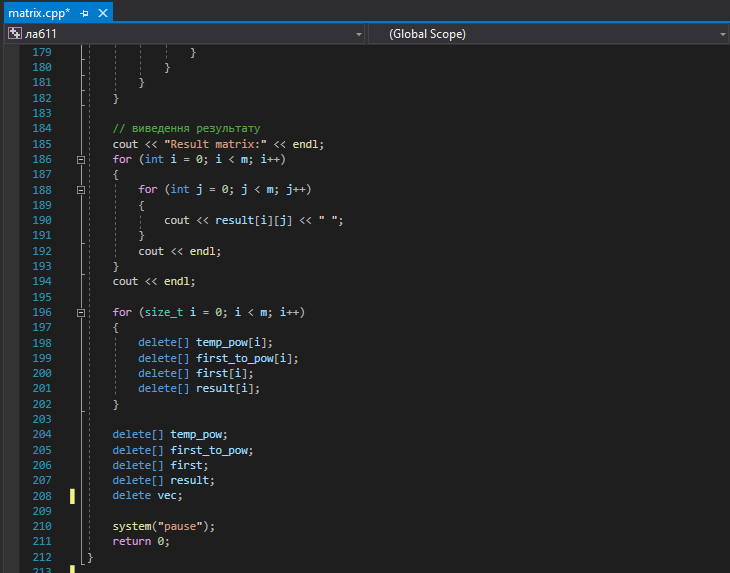


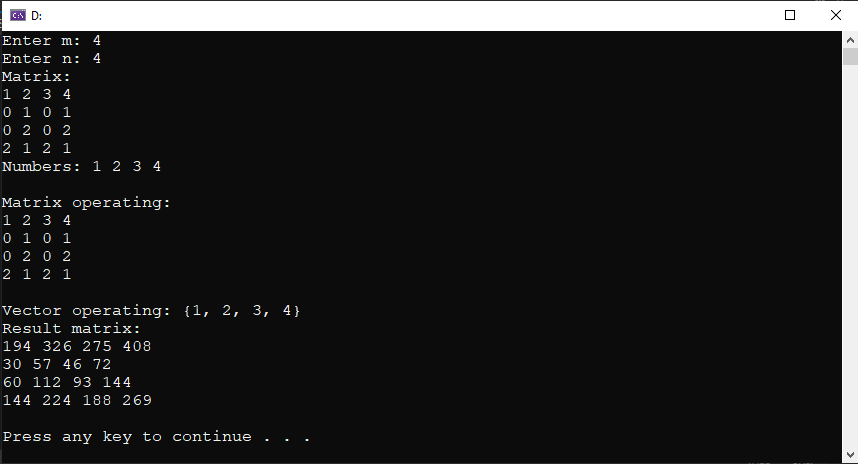












**Повний код:**

#include <iostream>

using namespace std;

int main()

{

int mm;

cout << "Enter m: ";

cin >> mm;

const int m = mm;

int nn;

cout << "Enter n: ";

cin >> nn;

const int n = nn;

//початкова матриця

cout << "Matrix: ";

int\*\* first = new int\* [m];

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

{

first[i] = new int[m];

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

{

int m1;

cin >> m1;

first[i][j] = m1;

}

}

// тимчасова матриця для зберігання матриці до степення

int\*\* temp\_pow = new int\* [m];

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

{

temp\_pow[i] = new int[m];

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

{

temp\_pow[i][j] = 0;

}

}

// матриця для зберігання матриці до степення

int\*\* first\_to\_pow = new int\* [m];

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

{

first\_to\_pow[i] = new int[m];

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

{

first\_to\_pow[i][j] = first[i][j];

}

}

cout << "Numbers: ";

int\* vec = new int[n];

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

{

int m1;

cin >> m1;

vec[i] = m1;

}

int\*\* result = new int\* [n];

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

{

result[i] = new int[n];

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

{

if (i == j)

{

result[i][j] = vec[0] + first[i][j] \* vec[1];

}

else

{

result[i][j] = first[i][j] \* vec[1];

}

}

}

cout << endl;

// матриця, за допомогою якої отримаємо кінцеву матрицю

cout << "Matrix operating:" << endl;

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

{

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

{

cout << first[i][j] << " ";

}

cout << endl;

}

cout << endl;

cout << "Vector operating: {";

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

{

cout << vec[i] << ", ";

}

cout << "\b\b}" << endl;

for (int k = 2; k < n; k++)

{

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

{

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

{

\_\_asm

{

mov edx, temp\_pow // edx = temp\_pow

mov esi, i

imul esi, 4

add edx, esi // edx = &temp\_pow[i]

mov edx, [edx]

mov esi, j

imul esi, 4

add edx, esi // edx = &temp\_pow[i][j]

push edx // заштовхуємо в стек

xor esi, esi // обнулення лічильника

mov ecx, n

\_start :

mov eax, first\_to\_pow

mov ebx, first

powered\_first\_to\_eax :

mov edx, i

mov eax, [eax + edx \* 4] // eax = a[i]

mov edx, esi

mov eax, [eax + edx \* 4] // eax = a[i][esi]

first\_to\_ebx :

mov edx, esi

mov ebx, [ebx + edx \* 4] // ebx = b[esi]

mov edx, j

mov ebx, [ebx + edx \* 4] // ebx = b[esi][j]

\_end :

pop edx // виштовхуємо зі стеку

imul eax, ebx // eax = a[i][esi] \* b[esi][j]

add[edx], eax

push edx // заштовхуємо в стек

inc esi // збільшуємо значення операнда

loop \_start // кінець циклу

pop edx // виштовхуємо зі стеку

}

}

}

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

{

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

{

\_\_asm

{

mov edx, temp\_pow // edx = temp\_pow

mov ecx, result // ecx = result

mov eax, first\_to\_pow // eax = first\_to\_pow

mov esi, i

imul esi, 4

add edx, esi // edx = &temp\_pow[i]

add eax, esi // eax = &first\_to\_pow[i]

add ecx, esi // ecx = &result[i]

mov edx, [edx]

mov ecx, [ecx]

mov eax, [eax]

mov esi, j

imul esi, 4

add edx, esi

add ecx, esi

add eax, esi

mov ebx, [edx]

mov[eax], ebx

mov esi, vec

mov edi, k

imul edi, 4

add esi, edi

mov esi, [esi]

imul ebx, esi

add[ecx], ebx

mov[edx], 0

}

}

}

}

// виведення результату

cout << "Result matrix:" << endl;

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

{

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

{

cout << result[i][j] << " ";

}

cout << endl;

}

cout << endl;

for (size\_t i = 0; i < m; i++)

{

delete[] temp\_pow[i];

delete[] first\_to\_pow[i];

delete[] first[i];

delete[] result[i];

}

delete[] temp\_pow;

delete[] first\_to\_pow;

delete[] first;

delete[] result;

delete vec;

system("pause");

return 0;

}

**Висновок:** під час виконання лабораторної роботи №11 я, використовуючи середовище Microsoft Visual Studio, написала програму опрацювання масивів даних на мові Assembler.