МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГАОУ ВО «Крымский федеральный университет имени В. И. Вернадского»
Физико-технический институт
Кафедра компьютерной инженерии и моделирования

Лабораторная работа №6
по курсу «Структуры и алгоритмы обработки данных»
на тему: «Действия над матрицами в блок-схемах»

Выполнил:
студент 1 курса
группы ПИ-б-о-233(1)
Иващенко Денис Олегович

Зачтено (100, 100)
17.12.2023

Проверила:
старший преподаватель
кафедры компьютерной
инженерии и моделирования
Горская И.Ю.

Симферополь, 2023

Лабораторная работа № 6

Тема:Действия над матрицами в блок-схемах

Цель работы: научиться действовать над матрицами в блок-схемах.

Перед выполнением лабораторной работы:

1. Были изучены теоретические сведения в методических указаниях к выполнению данной лабораторной работы; подробно рассмотрены приведенные практические примеры.

2. Прочитан соответствующий материал в электронном конспекте лектора по данному курсу.

В соответствии с индивидуальным заданием выполнены два задания.

17) Дана матрица A(n X n),элементы которой различны. Найти наибольший элемент среди стоящих на главной и побочной диагоналях и поменять его местами с элементом, стоящим на пересечении этих диагоналей.

1)Для начала подключим нужные библиотеки, а именно iostream- для ввода и вывода

2)создадим переменную, которая нам нужна, чтобы мы знали какой размеренности будет матрица. Также, чтобы мы могли внести числа в матрицу

3)Дальше мы вносим наши числа в матрицу с помощью цикла for

4)инициализирует две целочисленные переменные maxI, maxJ для хранения индексов максимального элемента на главной и второстепенной диагоналях матрицы.

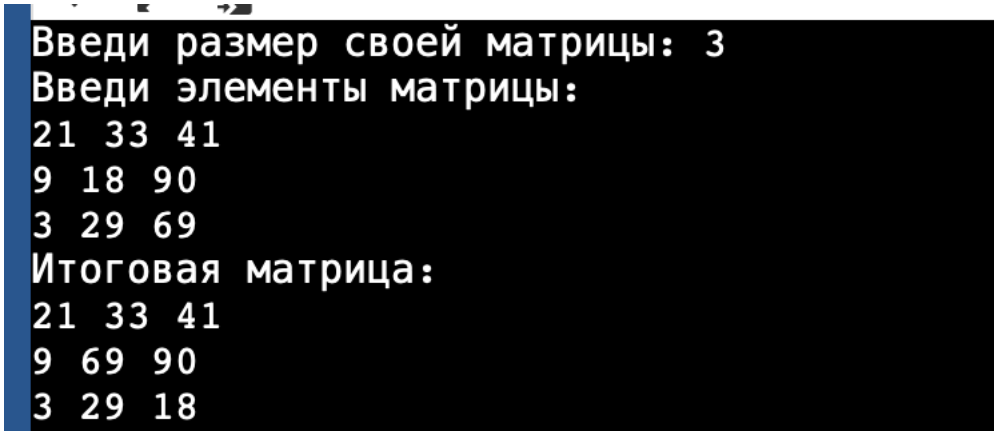
5)проверяем, больше ли элемент на главной диагонали текущего максимального элемента maxElement если значение true, он обновляет maxElement текущим элементом, maxI текущим индексом строки и maxJ текущим индексом столбца. Дальше проверяем больше ли элемент на дополнительной диагонали текущего максимального элемента maxElement

Если значение true, он обновляет maxElement текущим элементом, maxI текущим индексом строки и maxJ текущим индексом столбца. Дальше заменим максимальный элемент элементом на пересечении диагоналей, используя временную переменную temp

6) Дальше выводим результат с помощью цикла for

```cpp
#include <iostream>// для работы ввода и вывода
using namespace std;// для того, чтобы не писать std::

int main() {
    int n;// переменная n для того чтобы человек мог ввести размер матрицы
    cout << "Введи размер своей матрицы: ";
    cin >> n;//пользователь вводит

    int A[n][n];// матрица с размером, который поставит пользователь
    cout << "Введи элементы матрицы:  " << endl;
    // цикл по которому мы заполняем матрицу
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            cin >> A[i][j];
        }
    }
    //переменнеы maxI, maxJ
    int maxElement = A[0][0];
    int maxI = 0, maxJ = 0;

    // поиск большого числа на диагоналях
    for (int i = 0; i < n; i++) {
        if (A[i][i] > maxElement) {
            maxElement = A[i][i];
            maxI = i;
            maxJ = i;
        }
        if (A[i][n - 1 - i] > maxElement) {
            maxElement = A[i][n - 1 - i];
            maxI = i;
            maxJ = n - 1 - i;
        }
    }
```

```
    // перестановка элементов
    int temp = A[maxI][maxJ];
    A[maxI][maxJ] = A[n  - maxJ][n  - maxI];
    A[n  - maxJ][n  - maxI] = temp;

    // вывод нашей получившейся матрицы
    cout << "Итоговая матрица:" << endl;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            cout << A[i][j] << " ";
        }
        cout << endl;
    }

    return 0;
}
```



Результат программы:

5. По массиву X(n) построить матрицу $A(n \times n)$, где $a_{ij} = x_i \cdot x_j$, $i, j = 1, n$. Вычислить $\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}^2}$.

5)

1)Подключаем необходимые библиотеки для дальнейшей работы

2)Создадим две функции(printmat - выводит размер матрицы (n x m) с встроенной функцией setw() из библиотеки iomanip) printarray - выводит содержимое вектора array на экран.

3) Создадим две переменные(n - для размеренности массива, t- для вручного ввода чисел в матрицу), также создадим вектор. Потом мы эти переменные используем для того, чтобы пользователь мог ввести свои значения

4)Вводим числа в матрицу

```
    cin >> t;

    // здесь мы уже заполняем массив вручную
    if(t == 1){
        cout << "\nВведите элементы массива через пробел:" << endl;
        for(int i = 0; i < n; i++){
            cin >> arr[i];
        }
    }
    // создаем матрицу (двумерный массив)
    vector<vector<int>> matrix(n, vector<int>(n));
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            matrix[i][j] = arr[i] * arr[j];
        }
    }
    // вычисляется сумма квадратов всех элементов матрицы
    // с помощью вложенных циклов for и встроенной функции pow
    double sum = 0;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            sum += pow(matrix[i][j], 2);
        }
    }
//здесь уже вызываем нужные функции
    printarray(arr);
    printmat(n, n, matrix);
    // здесь выводится вычисленная сумма квадратов sqrt(cmath)
    cout << "\nОтвет = " << sqrt(sum) << endl;
}
```

5)Дальше вложенные циклы for заполняют элементы матрицы путем перемножения соответствующих элементов вектора arr

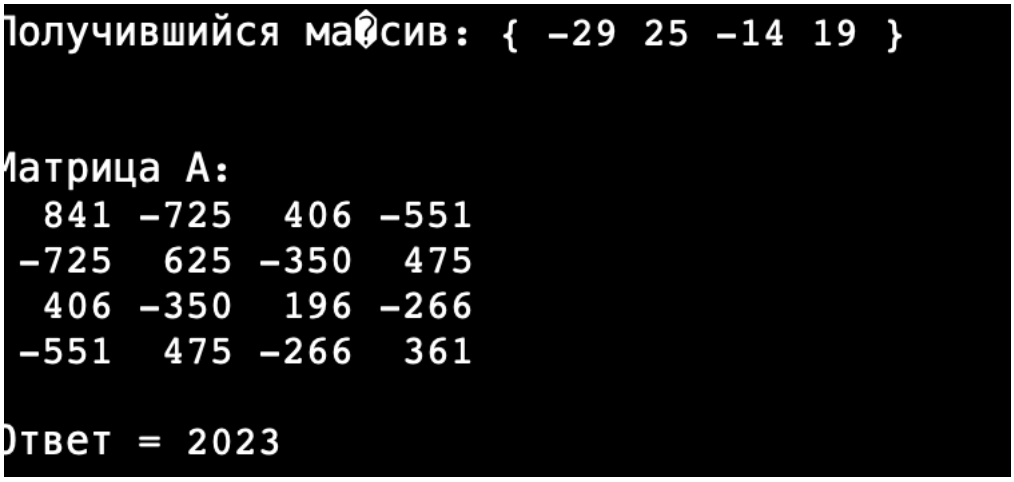
6) После, мы вычисляем сумму квадратов с помощью цикла for и встроенной функции pow() из библиотеки cmath

7)Вызываются функции, а также вывод суммы квадратов с встроенной

```cpp
#include <iostream>// для ввода и вывода
#include <vector>// для матрицы(динамический массив)
#include <cmath>// для возведения в квадрат
#include <iomanip>// для матрицы
#include <algorithm>//и еще раз для матрицы


using namespace std;// для того, чтобы не писать std::
// функция, которая выводит n x m. Также, здесь мы используем встроенную функцию setw() из библиотеки iomanip
void printmat(int n, int m, const vector<vector<int>>& matrix) {
    cout << "\nМатрица A: " << endl;
    for (size_t i = 0; i < n; i++) {
        for (size_t j = 0; j < m; j++) {
            cout << setw(5) << matrix[i][j];
        }
        cout << endl;
    }
}
// здесь уже вывод на экран(фунция)
void printarray(const vector<int>& array){
    cout << "\nПолучившийся массив: { ";
    for(const auto &elem: array){
        cout << elem << ' ';
    }
    cout << "}\n" << endl;
}
int main()
{
    // здесь мы уже объявляем переменные n, t для размерности массива
    int n, t;
    cout << "\nВведите n: " << endl;
    // какой размер массива
    cin >> n;
    // дианмический массив arr с размером n
    vector<int> arr(n);
    cout << "\nВыберите вариант (1 - заполинть массив вручную"<< endl;
    cin >> t;
```

функции sqrt() из библиотеки cmath

8) Вот результат нашей программы:



Вывод: сегодня на лабораторной работе по предмету «Структуры и алгоритмы обработки данных» я научился действовать над матрицами в блок-схемах.