**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное автономное образовательное

учреждение высшего образования

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

Инженерная школа информационных технологий и робототехники

Отделение информационных технологий

Направление информатика и вычислительная техника

Отчет

по лабораторной работе №4

по дисциплине

**«**Программирование**»**

Вариант №6

**Обработка многомерных динамических массивов**

Выполнил:

Студент группы 8В32 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Гребнева К. А.

Проверил:

Ассистент ОИТ ИШИТР \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Малкин А. Ю.

# Цель работы

Познакомиться с работой с динамической памятью.

# Задание

Упорядочить строки матрицы по возрастанию значений сумм их элементов.

# Ход работы

Мы создаем новый проект, и пишем код (его можно увидеть в приложении 1). Программа сначала запрашивает размерность матрицы, а потом просит ввести всю матрицу. Затем она обрабатывает матрицу и выводит конечную матрицу в консоль. На рисунке 1 можно увидеть результат работы программы для заданной пользователя матрицы.

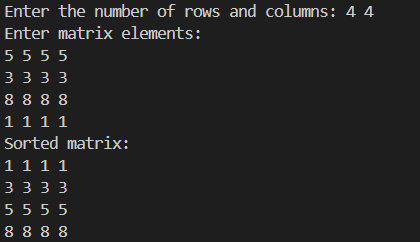


Рисунок 1 – Результат работы программы для заданной пользователем матрицы

# Вывод

В ходе лабораторной работы были получены навыки работы в С++, изучен принцип работы динамических массивов.

# Приложение 1 – Текст программы по упорядочиванию строк матрицы по возрастанию их сумм

#include <iostream>

using namespace std;

/\*\*

  \* @brief Читает матрицу

  \* @param matrix матрица которая будет заполнена

  \* @param rows количество рядов

  \* @param cols кол-во столбцов

\*/

void readMatrix(int\*\* matrix, int rows, int cols) {

    cout << "Enter matrix elements:" << endl;

    for (int i = 0; i < rows; i++) {

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

            cin >> matrix[i][j];

        }

    }

}

/\*\*

  \* @brief Нахождение суммы элементов

  \* @param rows количество рядов

  \* @param cols кол-во столбцов

  \* @return сумма строки матрицы

\*/

int rowSum(int\* row, int cols) {

    int sum = 0;

    for (int i = 0; i < cols; i++) {

        sum += row[i];

    }

    return sum;

}

/\*\*

  \* @brief Обмен местами 2х строк

  \* @param a первая строка

  \* @param b вторая строка

\*/

void swap(int\*& a, int\*& b) {

    int\* temp = a;

    a = b;

    b = temp;

}

/\*\*

  \* @brief Сортировка матрицы основываясь на сумме

  \* @param matrix матрица

  \* @param rows количество рядов

  \* @param cols кол-во столбцов

\*/

void sortRows(int\*\* matrix, int rows, int cols) {

    for (int i = 0; i < rows - 1; i++) {

        for (int j = 0; j < rows - i - 1; j++) {

            if (rowSum(matrix[j], cols) > rowSum(matrix[j + 1], cols)) {

                swap(matrix[j], matrix[j + 1]);

            }

        }

    }

}

/\*\*

  \* @brief Вывод матрицы на экран

  \* @param matrix матрица

  \* @param rows количество рядов

  \* @param cols кол-во столбцов

\*/

void displayMatrix(int\*\* matrix, int rows, int cols) {

    cout << "Sorted matrix:" << endl;

    for (int i = 0; i < rows; i++) {

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

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

        }

        cout << endl;

    }

}

int main() {

    int rows, cols;

    cout << "Enter the number of rows and columns: ";

    cin >> rows >> cols;

    // Выделение памяти под матрицу

    int\*\* matrix = new int\*[rows];

    for (int i = 0; i < rows; i++) {

        matrix[i] = new int[cols];

    }

    readMatrix(matrix, rows, cols);

    sortRows(matrix, rows, cols);

    displayMatrix(matrix, rows, cols);

    for (int i = 0; i < rows; i++) {

        delete[] matrix[i];

    }

    delete[] matrix;

    return 0;

}