Пермский национальный исследовательский политехнический университет

Кафедра информационные технологии и автоматизированные системы

Лабораторная работа №5

Функции и массивы

Вариант: 9

|  |  |
| --- | --- |
|  | Выполнил:  студент группы: ИВТ-24-2б  Ваулин Артём Сергеевич  Проверил:  доцент кафедры ИТАС  О.А. Полякова |

Пермь 2025

# Постановка задачи

Разработать программное средство, предназначенное для обработки набора квадратных матриц. Основная цель программы заключается в определении максимальной суммы элементов, расположенных строго ниже главной диагонали, среди заданного набора матриц. Программа должна принимать на вход количество матриц и их размерность, а также сами матрицы, и выводить в качестве результата максимальное значение указанной суммы.

# Анализ задачи

Для достижения поставленной цели необходимо выполнить следующие этапы:

1. **Получение входных данных:** Программа должна получить от пользователя информацию о количестве квадратных матриц для обработки и их общей размерности. Также требуется ввод элементов каждой матрицы.
2. **Вычисление суммы элементов ниже главной диагонали для каждой матрицы:** Для каждой введенной квадратной матрицы необходимо определить и просуммировать элементы, которые находятся строго ниже главной диагонали. Главная диагональ матрицы состоит из элементов, у которых индекс строки равен индексу столбца. Элементы, расположенные ниже главной диагонали, характеризуются индексом строки, превышающим индекс столбца.
3. **Поиск максимальной суммы:** После вычисления сумм элементов ниже главной диагонали для всех входных матриц, программа должна определить и сохранить максимальное значение из полученных сумм.
4. **Вывод результата:** Программа должна предоставить пользователю результат работы, а именно – максимальное значение суммы элементов, расположенных ниже главной диагонали, найденное среди всех обработанных матриц.

**Входные данные:**

* Количество квадратных матриц для обработки.
* Размерность квадратных матриц (предполагается, что все матрицы имеют одинаковую размерность).
* Набор квадратных матриц, представленных в виде своих элементов.

**Выходные данные:**

* Максимальное значение суммы элементов, расположенных ниже главной диагонали, среди всех входных матриц.

# Блок-схема

начало

int sumBelowDiagonal(int matrix[MAX\_SIZE][MAX\_SIZE], int size)

#include <iostream>

#include <cstdlib>

#include <ctime>

#include <clocale>

using namespace std;

const int MAX\_SIZE = 10;

int sum = 0;

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

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

if (i > j)

sum += matrix[i][j];

return sum;

{

1

{

}

0

{

}

1

0

int main()

setlocale(LC\_ALL, "Russian");

int n;

cout << "Введите количество матриц: ";

cin >> n;

int maxSize;

cout << "Введите размер квадратной матрицы (не более " << MAX\_SIZE << "): ";

cin >> maxSize;

if (maxSize <= 0 || maxSize > MAX\_SIZE)

cout << "Некорректный размер матрицы." << endl;

return 1;

A

{

1

0

{

}

A

srand(time(0));

int maxDiagonalSum = -1e9;

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

int matrix[MAX\_SIZE][MAX\_SIZE];

cout << "\nМатрица №" << matrixCount + 1 << ":" << endl;

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

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

matrix[i][j] = rand() % 100;

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

cout << endl;

B

C

D

{

{

1

0

1

1

{

0

0

}

}

B

int currentSum = sumBelowDiagonal(matrix, maxSize);

cout << "Сумма элементов ниже главной диагонали: " << currentSum << endl;

if (currentSum > maxDiagonalSum)

maxDiagonalSum = currentSum;

C

D

cout << "\nМаксимальная сумма элементов ниже главной диагонали среди всех матриц: " << maxDiagonalSum << endl;

return 0;

}

1

0

}

# Код программы

#include <iostream>

#include <cstdlib>

#include <ctime>

#include <clocale>

using namespace std;

const int MAX\_SIZE = 10;

// Функция для вычисления суммы элементов ниже главной диагонали

int sumBelowDiagonal(int matrix[MAX\_SIZE][MAX\_SIZE], int size)

{

int sum = 0;

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

{

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

{

if (i > j) // Элементы ниже главной диагонали: строка > столбец

{

sum += matrix[i][j];

}

}

}

return sum;

}

int main()

{

setlocale(LC\_ALL, "Russian");

int n; // Количество матриц

cout << "Введите количество матриц: ";

cin >> n;

int maxSize;

cout << "Введите размер квадратной матрицы (не более " << MAX\_SIZE << "): ";

cin >> maxSize;

if (maxSize <= 0 || maxSize > MAX\_SIZE)

{

cout << "Некорректный размер матрицы." << endl;

return 1;

}

srand(time(0));

int maxDiagonalSum = -1e9; // Инициализируем минимальным значением, чтобы найти максимум

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

{

int matrix[MAX\_SIZE][MAX\_SIZE];

cout << "\nМатрица №" << matrixCount + 1 << ":" << endl;

// Заполнение матрицы случайными числами и вывод на экран

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

{

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

{

matrix[i][j] = rand() % 100;

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

}

cout << endl;

}

int currentSum = sumBelowDiagonal(matrix, maxSize);

cout << "Сумма элементов ниже главной диагонали: " << currentSum << endl;

if (currentSum > maxDiagonalSum)

{

maxDiagonalSum = currentSum;

}

}

cout << "\nМаксимальная сумма элементов ниже главной диагонали среди всех матриц: " << maxDiagonalSum << endl;

return 0;

}

# Результат работы

