



Министерство науки и высшего образования Российской Федерации  
Калужский филиал  
федерального государственного бюджетного  
образовательного учреждения высшего образования  
«Московский государственный технический университет имени Н.Э. Баумана»  
(национальный исследовательский университет)  
(КФ МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУК «Информатика и управление»


КАФЕДРА ИУК4 «Программная инженерия»

## ЛАБОРАТОРНАЯ РАБОТА №1

«Разработка программного кода. Стандарт кодирования»

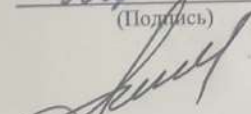
ДИСЦИПЛИНА: «Основы программной инженерии»

Выполнил: студент гр. ИУК4-22Б

  
(Подпись)

(\_\_\_\_\_) Моряков В.Ю. (\_\_\_\_\_) (Ф.И.О.)

Проверил:

  
(Подпись)

(\_\_\_\_\_) Амеличев Г.Э. (\_\_\_\_\_) (Ф.И.О.)

Дата сдачи (защиты):

22.03.2023

Результаты сдачи (защиты):

- Балльная оценка: (6)

- Оценка:

Калуга, 2023 г.



Министерство науки и высшего образования Российской Федерации  
Калужский филиал  
федерального государственного бюджетного  
образовательного учреждения высшего образования  
«Московский государственный технический университет имени Н.Э. Баумана  
(национальный исследовательский университет)»  
(КФ МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУК «Информатика и управление»

КАФЕДРА ИУК4 «Программная инженерия»

## ЛАБОРАТОРНАЯ РАБОТА №1

«Разработка программного кода. Стандарт кодирования»

ДИСЦИПЛИНА: «Основы программной инженерии»

Выполнил: студент гр. ИУК4-22Б \_\_\_\_\_ (\_\_\_\_Моряков В.Ю.\_\_\_\_)  
(Подпись) (Ф.И.О.)

Проверил: \_\_\_\_\_ (\_\_\_\_Амеличев Г.Э.\_\_\_\_)  
(Подпись) (Ф.И.О.)

Дата сдачи (защиты):

Результаты сдачи (защиты):

- Балльная оценка:

- Оценка:

Калуга, 2023 г.

**Цель:** формирование практических навыков следования стандарту кодирования при разработке программного обеспечения.

**Задачи:**

1. Разделиться на команды по 2 человека.
2. Изучив задание варианта разделить задачи между собой.
  3. Произвести рефакторинг кода, согласно стандарту кодирования.
  4. Составить таблицу идентификаторов.
  5. Подготовить отчет.

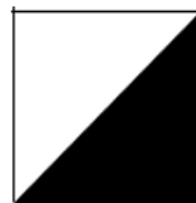
**Вариант 1**

Квадратную матрицу 16x16 заполнить случайными числами из диапазона [-9..9], вывести ее на экран.

а) В каждой строке матрицы найти произведение элементов, расположенных после максимального элемента в этой строке.

б) Проверить, симметричны ли все строки относительно среднего элемента.

в) Вычислить сумму элементов матрицы, выделенных чёрным цветом (матрица квадратная).



**Листинг программы**

```
#include <iostream>
#include<limits.h>
#include<ctime>

using namespace std;

const int MatrixSize{ 16 };

void createMatrix(int (&matrix)[MatrixSize][MatrixSize]);
void printMatrix(int (&matrix)[MatrixSize][MatrixSize]);
void fillMaximalIndexes(int (&matrix)[MatrixSize][MatrixSize], int
(&maxIndexElements)[MatrixSize]);
void calculateSumInRow(int (&matrix)[MatrixSize][MatrixSize], int
(&maxIndexElements)[MatrixSize], int (&sumOfRows)[MatrixSize]);
void pproductOfElementsAfterMaxElement(int (&sumOfRows)[16]);
bool checkingStringsForSymmetry(int matrix[][MatrixSize], const int
MatrixSize);
int calculateSumOfBlackMatrixElements(int matrix[][MatrixSize], const int
MatrixSize);

int main() {
    int matrix[MatrixSize][MatrixSize]{ {100, 100, 100, 1},{100, 100, 1, 1},
{100, 1, 1, 1}, {1, 1, 1, 1} };

    setlocale(LC_ALL, "ru");

    // a

    srand(time(NULL));
```

```

createMatrix(matrix);

printMatrix(matrix);
cout << '\n';

int maxIndexElements[MatrixSize] = {0};

fillMaximalIndexes(matrix, maxIndexElements);

int sumOfRows[MatrixSize] = {0};

calculateSumInRow(matrix, maxIndexElements, sumOfRows);

pproductOfElementsAfterMaxElement(sumOfRows);
cout << '\n';

// 6
bool stringsAreSymmetrical = checkingStringsForSymmetry(matrix,
MatrixSize);
if (stringsAreSymmetrical) {
    cout << "Все строки симметричны относительно среднего элемента.";
} else {
    cout << "Строки не симметричны относительно среднего элемента.";
}
cout << '\n';

// 8
cout << "Сумма элементов матрицы, выделенных чёрным цветом равна " <<
calculateSumOfBlackMatrixElements(matrix, MatrixSize) << ".\n" << endl;

return 0;
}

void createMatrix(int (&matrix)[MatrixSize][MatrixSize]) {
    for (int i = 0; i < MatrixSize; i++) {
        for (int j = 0; j < MatrixSize; j++) {
            matrix[i][j] = rand() % 19 + (-9);
        }
    }
}

void printMatrix(int (&matrix)[MatrixSize][MatrixSize]) {
    for (int i = 0; i < MatrixSize; i++) {
        for (int j = 0; j < MatrixSize; j++) {
            cout << matrix[i][j] << " ";
        }
        cout << endl;
    }
}

void fillMaximalIndexes(int (&matrix)[MatrixSize][MatrixSize], int
(&maxIndexElements)[MatrixSize]) {
    for (int i = 0; i < MatrixSize; i++) {
        int localMaximal = INT_MIN;
        int localIndex = 0;
        for (int j = 0; j < MatrixSize; j++) {
            if (matrix[i][j] > localMaximal) {
                localMaximal = matrix[i][j];
                localIndex = j;
            }
        }
    }
}

```

```

        maxIndexElements[i] = localIndex;
    }
}

void calculateSumInRow(int (&matrix)[MatrixSize][MatrixSize], int
(&maxIndexElements)[MatrixSize], int (&sumOfRows)[MatrixSize]) {
    for (int i = 0; i < MatrixSize; i++) {
        int localSum = 1;

        for (int j = maxIndexElements[i]; j < 16; j++) {
            if (j == maxIndexElements[i]) {
                continue;
            }
            localSum *= matrix[i][j];
        }
        sumOfRows[i] = localSum;
    }
}

void pproductOfElementsAfterMaxElement(int (&sumOfRows)[MatrixSize]) {
    for (int i = 0; i < MatrixSize; i++) {
        cout << "Row index: " << i << " Row mult: " << sumOfRows[i] << endl;
    }
}

bool checkingStringsForSymmetry(int matrix[][MatrixSize], const int
MatrixSize) {
    bool stringsAreSymmetrical = true;

    for (size_t i = 0; i < MatrixSize / 2; i++) {
        for (size_t j = 0; j < MatrixSize; j++) {
            if (matrix[i][j] != matrix[MatrixSize - 1 - i][j]) {
                stringsAreSymmetrical = false;
                break;
            }
        }

        if (!stringsAreSymmetrical) {
            break;
        }
    }

    return stringsAreSymmetrical;
}

int calculateSumOfBlackMatrixElements(int matrix[][MatrixSize], const int
MatrixSize) {
    int sumOfBlackMatrixElements{};

    for (size_t i = 0; i < MatrixSize; i++) {
        for (size_t j = i; j < MatrixSize; j++) {
            sumOfBlackMatrixElements += matrix[MatrixSize - 1 - i][j];
        }
    }

    return sumOfBlackMatrixElements;
}

```

## Результат работы программы

-7 6 -9 -5 -2 -7 -7 -9 1 1 0 4 6 4 -2 8  
1 1 2 7 3 -2 6 9 -1 1 2 6 -5 -7 -2 9  
8 3 1 8 4 -9 8 8 0 8 3 -9 3 -8 -1 6  
7 8 -3 -2 2 -8 -3 -9 -7 2 -3 6 6 3 9 0  
-4 -4 -5 -1 -1 9 -1 0 -1 -7 -5 3 5 4 1 -5  
-9 -7 2 -4 2 0 -9 3 -8 1 5 -6 -4 -8 -1 -5  
8 -4 -3 -1 8 -8 3 2 4 8 -7 -2 8 -5 9 3  
1 9 8 8 -4 3 -9 8 5 -3 6 2 8 -5 -4 7  
-2 5 2 -6 5 -5 -5 1 -2 7 3 0 -3 0 -1 -2  
3 -5 -1 1 -8 -4 0 1 3 -3 5 -5 7 2 4 5  
-4 4 3 -7 1 4 -8 2 6 -3 2 -4 -2 -4 -6 5  
0 -2 3 -9 -7 -4 1 -8 8 5 -7 -5 9 -6 -1 0  
7 3 6 -9 6 0 1 4 -7 -5 3 8 6 -7 -7 -1  
1 -1 4 -8 -4 -5 2 -3 7 -1 -6 1 4 2 6 3  
2 -6 8 -8 6 -4 0 3 5 -3 -6 8 -8 -9 -5 1  
8 9 -5 -9 2 -5 4 9 8 -6 8 6 4 9 -9 -6

Row index: 0 Row mult: 1  
Row index: 1 Row mult: 7560  
Row index: 2 Row mult: 0  
Row index: 3 Row mult: 0  
Row index: 4 Row mult: 0  
Row index: 5 Row mult: -960  
Row index: 6 Row mult: 3  
Row index: 7 Row mult: 1737228288  
Row index: 8 Row mult: 0  
Row index: 9 Row mult: 40  
Row index: 10 Row mult: -5760  
Row index: 11 Row mult: 0  
Row index: 12 Row mult: -294  
Row index: 13 Row mult: 864  
Row index: 14 Row mult: 0  
Row index: 15 Row mult: -455032832

Строки не симметричны относительно среднего элемента.  
Сумма элементов матрицы, выделенных чёрным цветом равна 51.

D:\learning\MGTU\2\_sem\ОПИ\lab1\x64\Release\lab1.exe (процесс 15460)  
завершил работу с кодом 0.

## Таблица идентификаторов

<b>Идентификаторы</b> – имена, используемые для переменных, функций, меток и других определяемых пользователем объектов. Могут состоять как из одного, так и из нескольких символов. Правила именования: В именах идентификаторов нужно использовать американский английский. Необходимо использовать Upper Camel casing или Lower Camel casing. Словосочетания должны быть набраны как одно слово.		
Имя идентификатора	Нотация	Условный перевод
maxIndexElements	Lower Camel Case	Индексы максимальных элементов в каждой строке
sumOfRows	Lower Camel Case	Сумма элементов в каждой строке по условию из задания
stringsAreSymmetrical	Lower Camel Case	Строки матрицы симметричные относительно центра
matrix	Lower Camel Case	Матрица
В идентификаторах недопустимо:		
1.Использование сокращений		
2.Использование транслита		
3.Венгерская нотация		
В идентификаторах следует избегать:		
1.Сленг		
2.Ключевые слова		
3.Разделение словосочетаний		
<b>Акронимы</b> – слова, образованные из начальных букв слов или словосочетаний. Правила именования: Акронимы, состоящие из трех или больше символов подчиняются рекомендациям для обычных слов. Для двух символов нужно использовать один из регистров.		
Акронимы	Нотация	Дешифровка
Dry	Upper Camel Case	Don't Repeat Yourself
В акронимах недопустимо:		
1.Использование всех букв в верхнем регистре		
2.Состоящих из двух символов, использовать верхний и нижний регистр вместе		
В акронимах следует избегать:		
1.Совместность акронимов		
<b>Именованние типов</b> Правила именования: Использовать нужно Upper Camel Case. Название записи должно быть существительным или именной группой.		
Типы	Нотация	Дешифровка
Container	Upper Camel Case	Контейнер
При объявлении типов недопустимо:		
1.Прибавление префиксов или суффиксов		
2.Использование общих слов		
<b>Процедуры и функции</b> – последовательность операторов, которая имеет имя, список параметров и может быть вызвана из различных частей программы. Функции, в отличие от процедур, в результате своего выполнения возвращают значение, которое может быть использовано в выражении. Правила именования: Нужно использовать стиль Upper Camel Case. Имена должны прояснять обязанности процедуры/функции и при этом быть максимально короткими. Правила создания: Подпрограммам обозначают имя, их параметры, и их тип.		
Процедура или функция	Нотация	Дешифровка

createMatrix printMatrix fillMaximalIndexes	Lower Camel Case Lower Camel Case Lower Camel Case	Создать матрицу Вывести матрицу Заполнить массив индексов максимальных элементов в каждой строке
calculateSumInRow	Lower Camel Case	Посчитать сумма элементов в каждой строке по условию из задания
productOfElementsAfterMaxElement	Lower Camel Case	Вывести элементы, находящиеся после максимальных элементов в каждой строке
checkingStringsForSymmetry	Lower Camel Case	Проверить симметричность матрицы относительно центра
calculateSumOfBlackMatrixElements	Lower Camel Case	Посчитать сумму элементов в выделенной черной области по условию
При именовании недопустимо:		
1.Использование слов как "List", "Array" для обозначение его множества		
2.Использование для обратных процедур пары антонимов		
При именовании следует избегать:		
1.Использование имен типов		
<b>Константы</b> – постоянные величины в ряду изменяющихся Правила именования Нужно использовать стиль Upper Camel case. Имена должны прояснять обязанности процедуры/функции и при этом быть максимально короткими. Правила использования Константы используется в тех случаях, где само значение не должно изменяться в ходе работы программы.		
Константы	Нотация	Дешифровка
MatrixSize	Upper Camel Case	Размер матрицы
<b>Локальные переменные</b> – переменные, используемые в определенном участке программы. Правила именования: Используется Lower Camel Case. Идентификаторы переменных должны представлять собой существительные или именные группы.		
Локальные переменные	Нотация	Дешифровка
stringsAreSymmetrical	Lower Camel Case	Строки матрицы симметричные относительно центра
localMaximal	Lower Camel Case	Максимальный элемент в итерации по строке
localIndex	Lower Camel Case	Индекс максимального элемента в итерации
localSum	Lower Camel Case	Сумма полученная в итерации по строке
sumOfBlackMatrixElements	Lower Camel Case	Сумма элементов в выделенной черной области по условию

**Вывод:** в ходе выполнения лабораторной работы сформированы практические навыки следования стандарту кодирования при разработке программного обеспечения.