

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра «Вычислительная техника»

**ОТЧЕТ**

По лабораторной работе №2  
«Оценка времени выполнения программ.»  
По дисциплине «Л и ОА в ИЗ»

Выполнили: ст. гр. 22ВВ4  
Жуков Илья  
Чумаев Сабит

Приняли: Юрова О.В.  
Акифьев И.В.

## **Цель работы:**

Написать код программы, выполнив следующие задания:

### **Задание 1:**

- 1.Вычислить порядок сложности программы (O-символику).
- 2.Оценить время выполнения программы и кода, выполняющего перемножение матриц, используя функции библиотеки `time.h` для матриц размерами от 100, 200, 400, 1000, 2000, 4000, 10000.
- 3.Построить график зависимости времени выполнения программы от размера матриц и сравнить полученный результат с теоретической оценкой.

### **Задание 2:**

- 1.Оценить время работы каждого из реализованных алгоритмов на случайном наборе значений массива.
- 2.Оценить время работы каждого из реализованных алгоритмов на массиве, представляющем собой возрастающую последовательность чисел.
- 3.Оценить время работы каждого из реализованных алгоритмов на массиве, представляющем собой убывающую последовательность чисел.
- 4.Оценить время работы каждого из реализованных алгоритмов на массиве, одна половина которого представляет собой возрастающую последовательность чисел, а вторая, – убывающую.
- 5.Оценить время работы стандартной функции `qsort`, реализующей алгоритм быстрой сортировки на выше указанных наборах данных.

## **Ход работы:**

### **Описание кода программы по заданию 1:**

В данной программе производится умножение двух случайно сгенерированных матриц различных размеров. Затем измеряется время выполнения этой операции для каждого размера матрицы из заданного списка.

```
static void Main()
{

//объявили массив matrixSizes, который содержит размеры матриц, для
которых будет выполняться умножение.
    int[] matrixSizes = { 100, 200, 400, 1000, 2000, 4000,
10000 };
```

```

//Перебираем данный массив
    foreach (int size in matrixSizes)
    {

//вызов метода GenerateRandomMatrix для генерации случайных матриц
(matrix1 и matrix2) заданного размера size x size.
        int[,] matrix1 = GenerateRandomMatrix(size, size);
        int[,] matrix2 = GenerateRandomMatrix(size, size);

//создание объекта класса Stopwatch для замера времени выполнения.
        Stopwatch stopwatch = new Stopwatch();

//запуск секундомера.
        stopwatch.Start();

//вызов метода MultiplyMatrices для умножения матриц matrix1 и matrix2
и сохранения результата в переменную resultMatrix.
        int[,] resultMatrix = MultiplyMatrices(matrix1, matrix2);

//остановка секундомера.
        stopwatch.Stop();

//сохранение прошедшего времени в переменную elapsedTime типа TimeSpan.
        TimeSpan elapsedTime = stopwatch.Elapsed;

//вывод сообщения о размере матрицы на консоль.
        Console.WriteLine($"Matrix size: {size}x{size}");

//вывод сообщения о времени выполнения на консоль.
        Console.WriteLine($"Elapsed                               Time:
{elapsedTime.TotalSeconds} seconds");
    }
}

//объявление метода GenerateRandomMatrix, который генерирует случайную
матрицу заданного размера rows x cols и возвращает ее.
    static int[,] GenerateRandomMatrix(int rows, int cols)
    {
        Random random = new Random();

//объявление двумерного массива matrix заданного размера rows x cols.
        int[,] matrix = new int[rows, cols];

//начало цикла for, который проходит по строкам матрицы.
        for (int i = 0; i < rows; i++)
        {

//начало цикла for, который проходит по столбцам матрицы.
            for (int j = 0; j < cols; j++)
            {

```

```

//генерация случайного числа от 0 до 99 и присваивание его элементу
матрицы с индексами i и j.
        matrix[i, j] = random.Next(100);
    }
}

return matrix;
}

//объявление метода MultiplyMatrices, который умножает две матрицы
matrix1 и matrix2 и возвращает полученную матрицу.
static int[,] MultiplyMatrices(int[,] matrix1, int[,] matrix2)
{
//получение количества строк в матрице
    int rows1 = matrix1.GetLength(0);

//получение количества столбцов в матрице
    int cols1 = matrix1.GetLength(1);

//получение количества столбцов в матрице
    int cols2 = matrix2.GetLength(1);

//объявление двумерного массива resultMatrix размером rows1 x cols2 для
сохранения результата умножения.
    int[,] resultMatrix = new int[rows1, cols2];

//начало цикла for, который проходит по строкам матрицы resultMatrix.
    for (int i = 0; i < rows1; i++)
    {

//начало цикла for, который проходит по столбцам матрицы resultMatrix.
        for (int j = 0; j < cols2; j++)
        {

//начало цикла for, который проходит по столбцам матрицы matrix1 или
строкам матрицы matrix2.
            for (int k = 0; k < cols1; k++)
            {

//Вычисление матрицы путем суммирования произведений соответствующих
элементов матриц matrix1 и matrix2.
                resultMatrix[i, j] += matrix1[i, k] * matrix2[k,
j];
            }
        }
    }

    Console.Write("-----\n");
}

```

```

        return resultMatrix;
    }

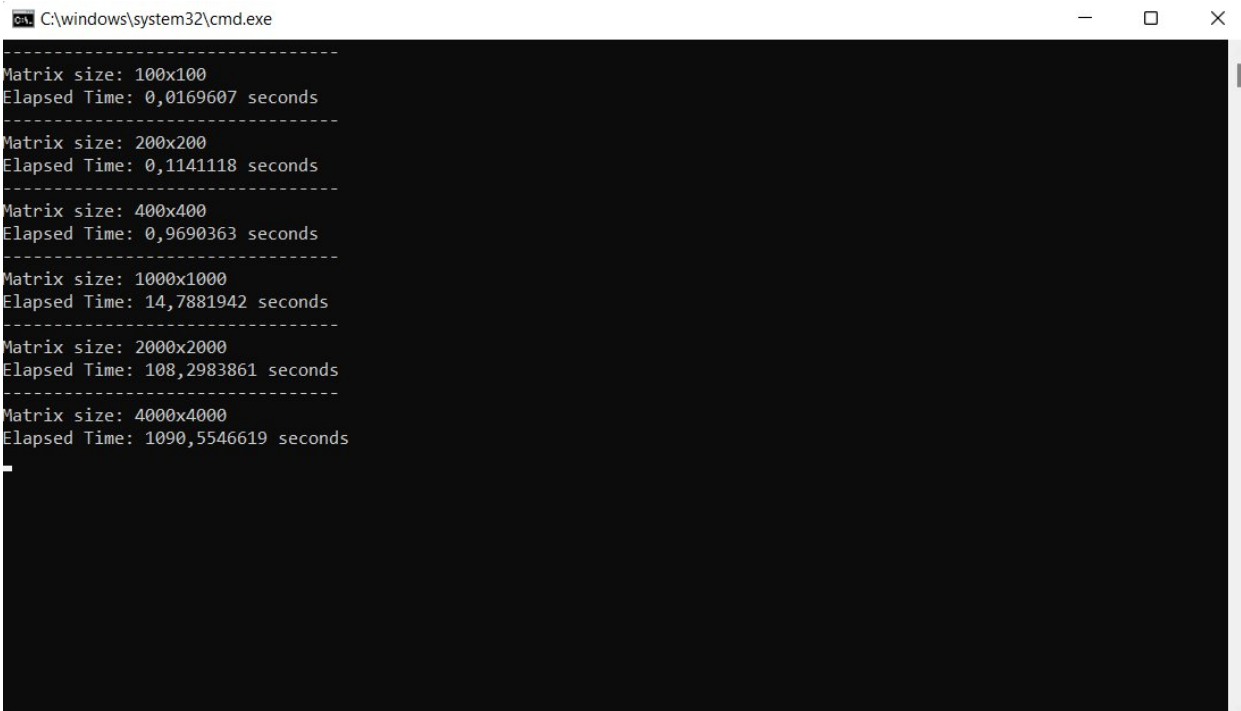
```

Порядок сложности перемножения матриц размера `NxN` составляет ` $O(N^3)$ `. Однако, в реальности время выполнения может отличаться из-за оптимизаций и особенностей компьютера.

sizeM	100	200	400	1000	2000	4000	10000
t	0,02	0,11	0,97	14,79	108,3	1090,55	30000



## Результат работы программы:



A screenshot of a Windows command prompt window titled "C:\windows\system32\cmd.exe". The window has a black background with white text. The text shows the results of a program's execution for different matrix sizes, with each result separated by a dashed line. The results are as follows:

Matrix size	Elapsed Time
100x100	0,0169607 seconds
200x200	0,1141118 seconds
400x400	0,9690363 seconds
1000x1000	14,7881942 seconds
2000x2000	108,2983861 seconds
4000x4000	1090,5546619 seconds