МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт информационных технологий

Кафедра «Информационные системы»

**ОТЧЕТ**

о выполнении лабораторной работы №6

по дисциплине “Тестирование Программного Обеспечения”

Выполнил: ст. гр. ИС/б-20-2-о

Белик Г. М.

Проверил: доц. каф.

«Информационные системы»

Строганов В. А.

Севастополь

2022

**ЛАБОРАТОРНАЯ РАБОТА №6**

**«Исследование способов профилирования программного обеспечения»**

**Цель работы**

Исследовать критические по времени выполнения участки программного кода и возможности их устранения. Приобрести практические навыки анализа программ с помощью профайлера EQATECProfiler.

**Задачи**

1. Разработать программу на основе библиотеки классов, реализованной и протестированной в предыдущей работе. Программа должна как можно более полно использовать функциональность класса. При необходимости для наглядности профилирования в методы класса следует искусственно внести задержку выполнения.
2. Выполнить профилирование разработанной программы, выявить функции, на выполнение которых тратится наибольшее время.
3. Модифицировать программу с целью оптимизации времени выполнения.
4. Выполнить повторное профилирование программы, сравнить новые результаты и полученные ранее, сделать выводы.

**Ход работы**

Задача состоит в нахождении ответа, либо выводе информации о наличии отрицательных элементов. Для анализа программы был выбран класс Matrix, а конкретно его функция Print для печати результата на экран произведения элементов в строках матрицы так как она занимает наибольшее время выполния. Способом совершенствования программы было найдено модернизацией решения путем переписания метода вывода.

Ниже приведен пример выполнения первоначальной программы и занимаемое время выполнения : инклюзивное 13,06 и эксклюзивное 3,38 (рисунок 1).

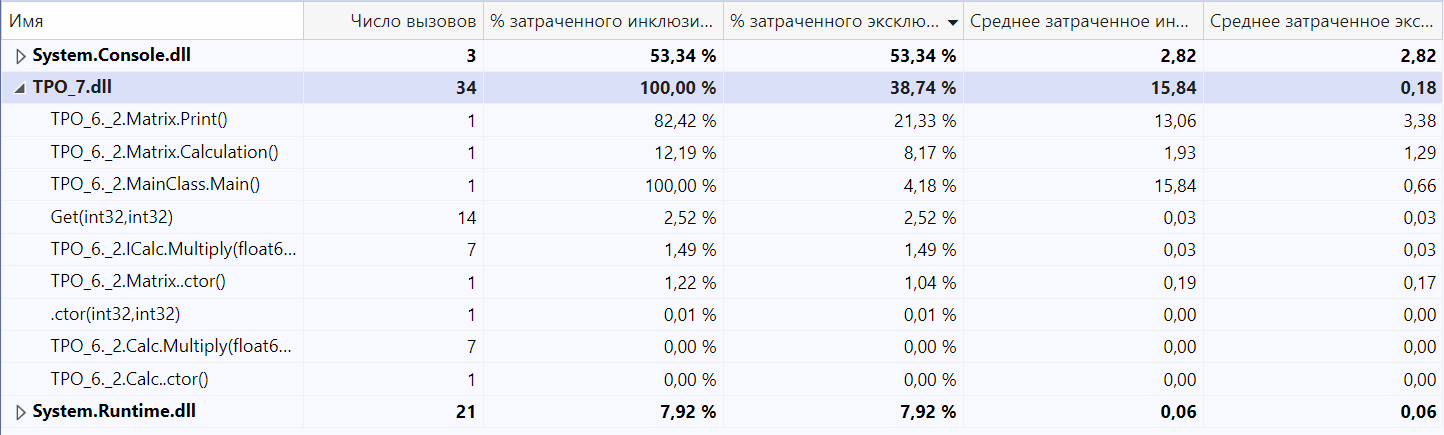


Рисунок 1 – Таблица занимаемого времени первого запуска

Далее приведен вариант выполнения модернизированной программы и время

выполнения : инклюзивное 7,71 и эксклюзивное 1,64 (рисунок 2).

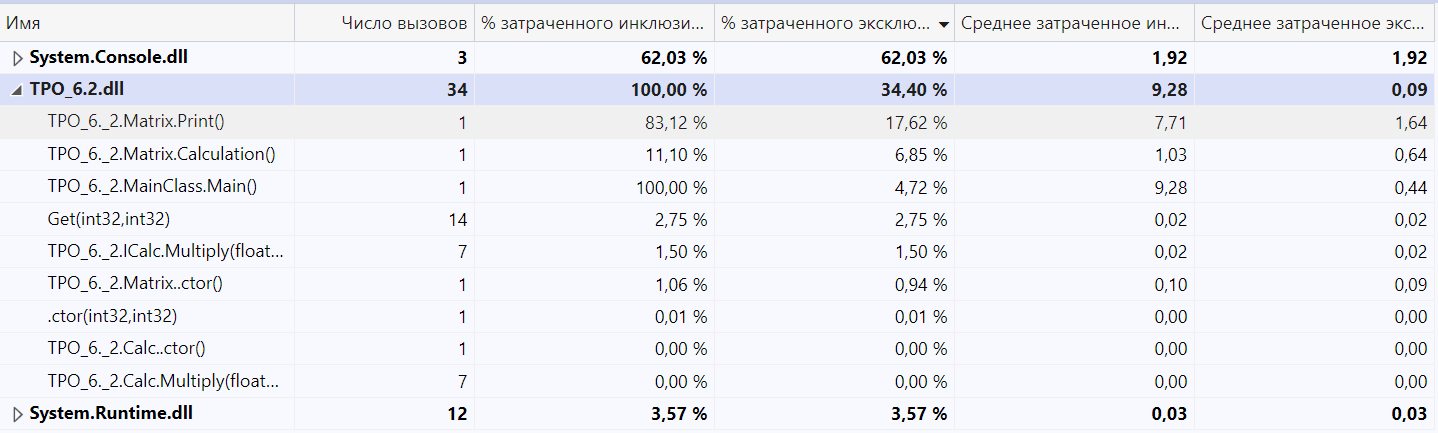


Рисунок 2 – Таблица занимаемого времени модернизированной программы

В ходе сравнения результатов можно заметить более быстрое выполнение

отдельного метода Print почти в 2 раза, и в целом класса Main соответственно. В результате модификации программы получаем оптимизацию времени выполнения.

Затем показан пример правильности и идентичности выполнения обеих вариантов

программ, чтобы удостовериться в том, что модернизация не повлияло на правильность

(рисунок 3).

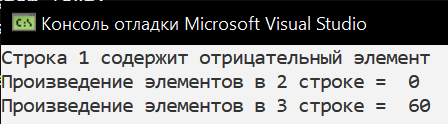


Рисунок 3 – Правильность реализации программы

**Текст программы класса MainClass:**

namespace TPO\_6.\_2

{

public class MainClass

{

public static void Main()

{

Matrix matrr = new Matrix();

matrr.Calculation();

matrr.Print();

}

}

}

**Текст программы Matrix:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace TPO\_6.\_2

{

public class Matrix

{

public int[] mass\_result = new int[3]{ 1, 1, 1 };

public int[,] matr = new int[3, 3] { { -1, 1, 2 }, { 0, 1, 2 }, { 3, 4, 5 } };

public void Print()

{

for (int i = 0; i < mass\_result.Length; i++)

{

if (mass\_result[i] >= 0)

Console.WriteLine($"Произведение элементов в {i + 1} строке = {mass\_result[i]}");

else

Console.WriteLine($"Строка {i+1} содержит отрицательный элемент");

}

}

public int[] Calculation()

{

ICalc calc = new Calc();

for (int i = 0; i < 3; i++)

for (int j = 0; j < 3; j++)

{

mass\_result[i] = (int)calc.Multiply(mass\_result[i], matr[i, j]);

if (matr[i, j] < 0)

{

mass\_result[i] = -1;

break;

}

}

return mass\_result;

}

}

}

**Текст программы ICalc:**

namespace TPO\_6.\_2

{

public interface ICalc

{

public double Multiply(double a, double b);

}

}

**Текст программы Calc:**

namespace TPO\_6.\_2

{

public class Calc : ICalc

{

public double Multiply(double a, double b)

{

return a \* b;

}

}

}

**Модернизированный метод Print (в классе Matrix):**

public void Print()

{

for (int i = 0; i < mass\_result.Length; i++)

{

if (mass\_result[i] >= 0)

Console.WriteLine("Произведение элементов в" + i +"строке = " + mass\_result[i]);

else

Console.WriteLine("Строка " +i+ " содержит отрицательный элемент");

}

}

**Выводы**

В ходе лабораторной работы была исследована возможность устранения

критических точек программного кода. Также были сделаны выполнения сделаны выводы о пользе данного метода. В результате выполнения работы были приобретены практические навыки анализа программ с помощью профайлера, выявить функции, на выполнение которых тратится наибольшее время, модификации с целью оптимизации времени выполнения.