Лабораторная работа №11
Тестирование производительности ПО

1 Цель работы

1.1 Освоить процесс профилирования ПО

1.2 Изучить методы тестирования производительности ПО

2 Литература

2.1 Игнатьев, А. В. Тестирование программного обеспечения : учебное пособие для вузов / А. В. Игнатьев. — 4-е изд., стер. — Санкт-Петербург : Лань, 2025. — 56 с. — ISBN 978-5-507-50858-7. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/481331 (дата обращения: 21.05.2025). — Режим доступа: для авториз. пользователей.

3 Подготовка к работе

3.1 Повторить теоретический материал (см. п.2).

3.2 Изучить описание лабораторной работы.

4 Основное оборудование

4.1 Персональный компьютер.

5 Задание

5.1 Тестирование производительности методов при помощи Benchmark

5.1.1 Создать новый проект консольного приложения

5.1.2 Установить Nuget-пакет BenchmarkDotNet

5.1.3 Создать новый публичный класс PrimesBenchmark с атрибутом [MemoryDiagnoser], в него добавить 2 статических метода:

```
private static bool IsPrimeNaive(int n)
{
    for (int i = 2; i < n; i++)
        if (n % i == 0) return false;
    return true;
}

private static bool IsPrimeFast(int n)
{
    if (n < 2) return false;
    if (n == 2) return true;
    if (n % 2 == 0) return false;
    int sqrt = (int)Math.Sqrt(n);
    for (int i = 3; i <= sqrt; i += 2)
        if (n % i == 0) return false;
    return true;
}
```

5.1.4 Добавить в класс публичное целочисленное поле N. Укажите для него атрибут [Params(10_000, 50_000, 100_000)] для тестирования работы методов на различных входных данных

5.1.5 Добавить в класс 2 публичных метода без параметров – один

вычисляет сумму простых чисел от 2 до n при помощи метода IsPrimeNaive, второй – при помощи метода IsPrimeFast.

5.1.6 Для методов указать атрибут [Benchmark], причем для первого указать параметр атрибута Baseline = true, чтобы производительность вычислялась относительно него.

5.1.7 Изменить конфигурацию сборки с Debug на Release

5.1.8 Запустить бенчмарк в основной программе при помощи BenchmarkRunner.Run<PrimesBenchmark>();

5.1.9 Дождаться завершения бенчмарка, сравнить полученные показатели для двух методов на различных наборах данных.

5.2 Профилирование производительности методов

5.2.1 Создать новый проект консольного приложения

```
class Program
{
    static void Main()
    {
        const string filePath = "server.log";
        if (!File.Exists(filePath))
            File.WriteAllLines(filePath,
GenerateFakeLogs(50_000));

        var sw = Stopwatch.StartNew();
        var result = AnalyzeLogs(filePath);
        sw.Stop();

        Console.WriteLine($"Ошибок:          {result.errorCount},
предупреждений: {result.warningCount}");
        Console.WriteLine($"Время                    выполнения:
{sw.ElapsedMilliseconds} мс");
    }


    static (int errorCount, int warningCount) AnalyzeLogs(string
path)
    {
        var lines = File.ReadAllLines(path);

        int errorCount = lines.Count(l => l.Contains("ERROR"));
        int      warningCount      =      lines.Count(l      =>
l.Contains("WARNING"));

        return (errorCount, warningCount);
    }

    static IEnumerable<string> GenerateFakeLogs(int count)
    {
        var rand = new Random();
        string[] types = { "INFO", "WARNING", "ERROR" };
        for (int i = 0; i < count; i++)
        {
            yield        return        $"{DateTime.Now:HH:mm:ss}
```

```
[{types[rand.Next(types.Length)]}] Message {i}";
        }
    }
}
```

5.2.2 Изменить конфигурацию сборки с Debug на Release

5.2.3 При помощи меню Отладка – Профилировщик производительности запустить отладку с опцией «Использование ЦП»

5.2.4 Оценить время выполнения кода, определить узкое место в методе.

5.2.5 Оптимизировать работу метода и выполнить профилирование повторно, сравнить результаты.

6 Порядок выполнения работы

6.1 Повторить теоретический материал п. 3.1;

6.2 Выполнить задания п.5.1-5.2

6.3 Ответить на контрольные вопросы п.8;

6.4 Заполнить отчет п. 7.

7 Содержание отчета

7.1 Титульный лист;

7.2 Цель работы;

7.3 Данные по ходу профилирования

7.4 Ответы на контрольные вопросы п. 6.3;

7.5 Вывод по проделанной работе.

8 Контрольные вопросы

8.1 Как запустить профилировщик производительности в Visual Studio?

8.2 Как просмотреть использование ЦП в профилировщике?

8.3 Как использовать BenchmarkDotNet?