**Паттерны проектирования**

**Задание 1**

**Условие:** В каждой задаче необходимо создать **один единственный экземпляр** класса, ограничив его создание через приватный конструктор и предоставляя доступ через статический метод.

**Логгер (Logger)**

Создать класс Logger, который хранит логи работы приложения в единственном экземпляре.

* Метод Log(string message): добавляет сообщение в общий список логов.
* Метод ShowLogs(): выводит все логи.
* Использовать Singleton, чтобы во всей программе существовал только **один логгер**.

**Код**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Task1

{

internal class Program

{

static void Main(string[] args)

{

Logger logger = Logger.Instance;

logger.Log("Приложение запущено");

logger.Log("Приложение работает");

logger.Log("Приложение завершено");

logger.ShowLogs();

}

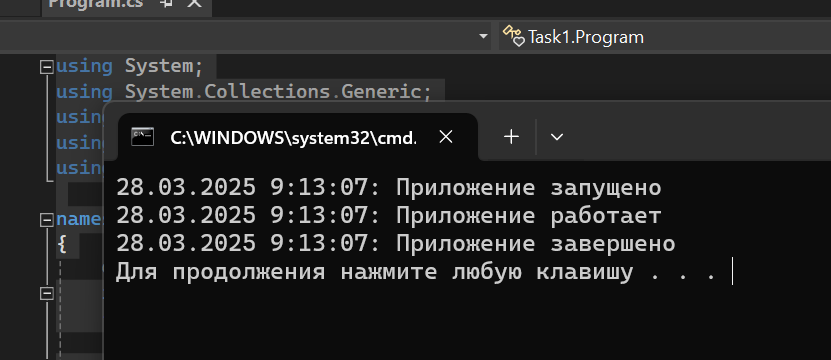
}

}

**Таблица 11.1 – Входные и выходные данные**

|  |  |
| --- | --- |
| **Входные данные** | **Выходные данные** |
|  | **28.03.2025 08:44:55: Приложение запущено**  **28.03.2025 08:44:55: Приложение работает**  **28.03.2025 08:44:55: Приложение завершено** |

**Анализ результата**

****

**Задание 2**

**2.1 Сортировка массива**

* + ArraySorter (контекст) выполняет сортировку.
  + ISortingStrategy (интерфейс) описывает метод Sort(int[] array).
  + BubbleSort, QuickSort, MergeSort (конкретные стратегии).

**Код**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Task2\_1

{

internal class Program

{

static void Main(string[] args)

{

int[] array = { 5, 3, 8, 6, 2 };

ArraySorter sorter = new ArraySorter(new BubbleSort());

sorter.SortArray(array);

Console.WriteLine("Bubble Sort: " + string.Join(", ", array));