# **10 Паттерны программирования**

Задание 1. Реализация паттерна Singleton. Разработать ConfigManager, который хранит настройки приложения. Метод SetConfig(string key, string value): задаёт значение настройки. Метод GetConfig(string key): получает значение настройки. Единственный экземпляр конфигурации должен существовать на всё приложение.

Листинг программы:

using System;

using System.Collections.Generic;

namespace E

{

public sealed class ConfigManager

{

private static readonly Lazy<ConfigManager> \_ic =

new Lazy<ConfigManager>(() => new ConfigManager());

private readonly Dictionary<string, string> \_confs = new Dictionary<string, string>();

private ConfigManager()

{

}

public static ConfigManager С => \_ic.Value;

public void SetConfig(string key, string value)

{

if (string.IsNullOrEmpty(key))

{

throw new ArgumentException("Ключ не может быть пустым.", nameof(key));

}

\_confs[key] = value;

}

public string GetConfig(string key)

{

if (string.IsNullOrEmpty(key))

{

throw new ArgumentException("Ключ не может быть пустым.", nameof(key));

}

return \_confs.TryGetValue(key, out string value) ? value : null;

}

}

class S

{

public static void Main(string[] args)

{

ConfigManager conf = ConfigManager.С;

conf.SetConfig("PracticalAssignment", "1.");

string? a = conf.GetConfig("PracticalAssignment");

Console.WriteLine($"Practical Assignment: {a}");

}

}

}

Таблица 10.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| PracticalAssignment, 1. | Practical Assignment: 1. |

Анализ результатов:

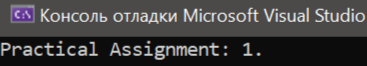


Рисунок 10.1 – Результат работы программы