Министерство образования Республики Беларусь

Учреждение образования

«Брестский Государственный технический университет»

Кафедра ИИТ

Лабораторная работа №4

По дисциплине «Современные платформы программирования»

Выполнил:

Студент 3 курса

Группы ПО-8

Лобарев А.М.

Проверил:

Крощенко А.А.

Брест 2024

Цель работы: приобрести практические навыки в области объектно-ориентированного проектирования.

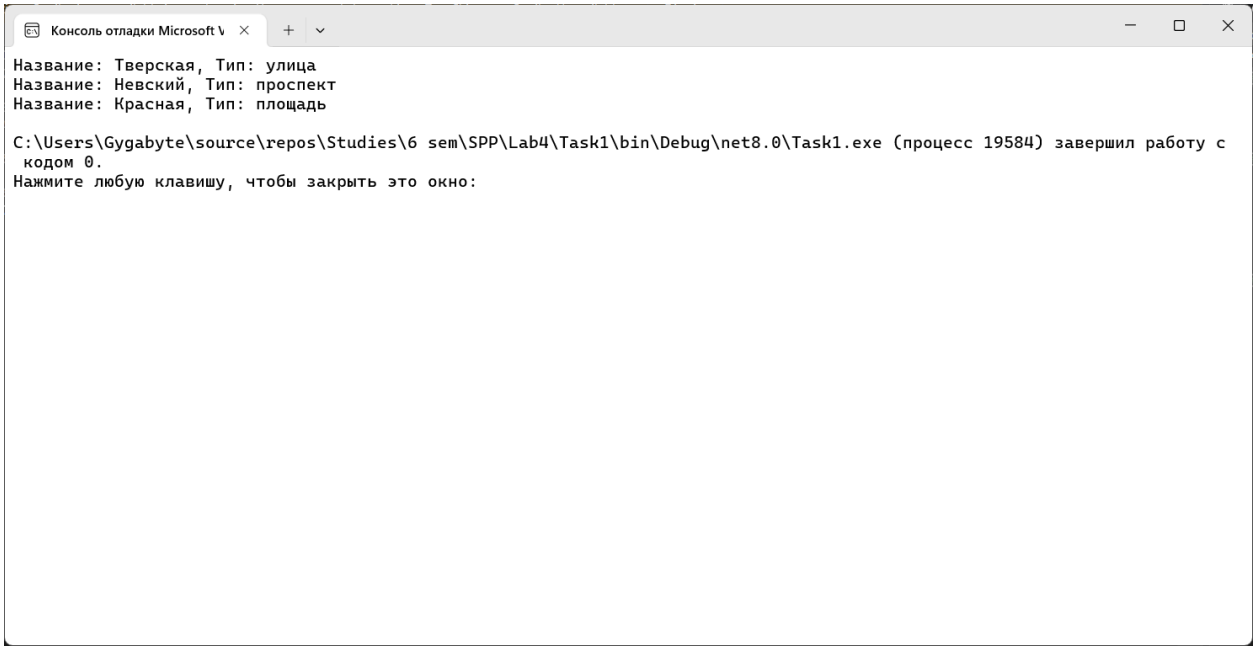
Вариант 16

Задание 1

Создать класс City (город) с внутренним классом, с помощью объектов которого можно хранить информацию о проспектах, улицах, площадях.

Код:

```
internal class City
{
    private List<Street> streets = new List<Street>();

    public class Street
    {
        public string Name { get; set; }
        public string Type { get; set; } // Тип, например: "улица", "проспект",
"площадь"

        public Street(string name, string type)
        {
            Name = name;
            Type = type;
        }
    }

    public void AddStreet(string name, string type)
    {
        streets.Add(new Street(name, type));
    }

    public void DisplayStreets()
    {
        foreach (var street in streets)
        {
            Console.WriteLine($"Название: {street.Name}, Тип: {street.Type}");
        }
    }
}



internal class Program
{
    static void Main()
    {
        City city = new City();
        city.AddStreet("Тверская", "улица");
        city.AddStreet("Невский", "проспект");
        city.AddStreet("Красная", "площадь");

        city.DisplayStreets();
    }
}
```

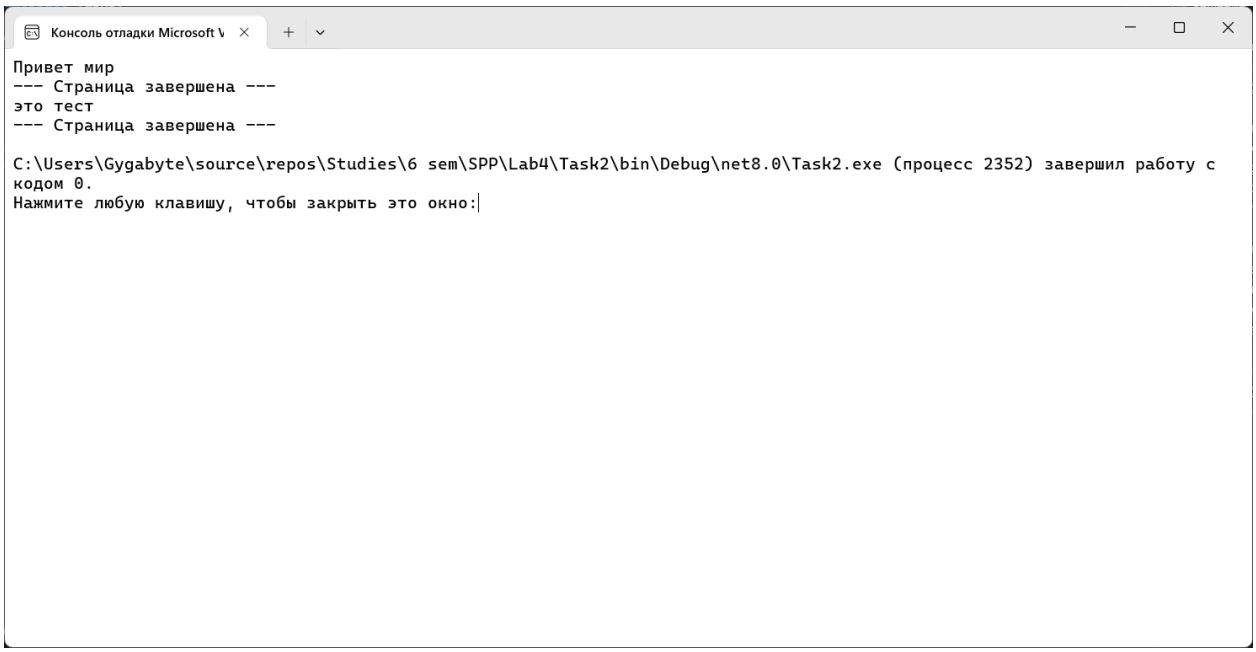


Задание 2

Создать класс Текст, используя классы Страница, Слово.

```
public class Word
{
    public string Value { get; private set; }

    public Word(string value)
    {
        Value = value;
    }
}



public class Page
{
    private List<Word> words = new List<Word>();

    public void AddWord(Word word)
    {
        words.Add(word);
    }

    public void PrintWords()
    {
        foreach (var word in words)
        {
            Console.Write(word.Value + " ");
        }
        Console.WriteLine();
    }
}
```

```
public class Text
{
    private List<Page> pages = new List<Page>();

    public void AddPage(Page page)
    {
        pages.Add(page);
    }

    public void PrintText()
    {
        foreach (var page in pages)
        {
            page.PrintWords();
            Console.WriteLine("--- Страница завершена ---");
        }
    }
}



internal class Program
{
    static void Main()
    {
        Word word1 = new Word("Привет");
        Word word2 = new Word("мир");
        Word word3 = new Word("это");
        Word word4 = new Word("тест");

        Page page1 = new Page();
        page1.AddWord(word1);
        page1.AddWord(word2);

        Page page2 = new Page();
        page2.AddWord(word3);
        page2.AddWord(word4);

        Text text = new Text();
        text.AddPage(page1);
        text.AddPage(page2);

        text.PrintText();
    }
}
```



Задание 3

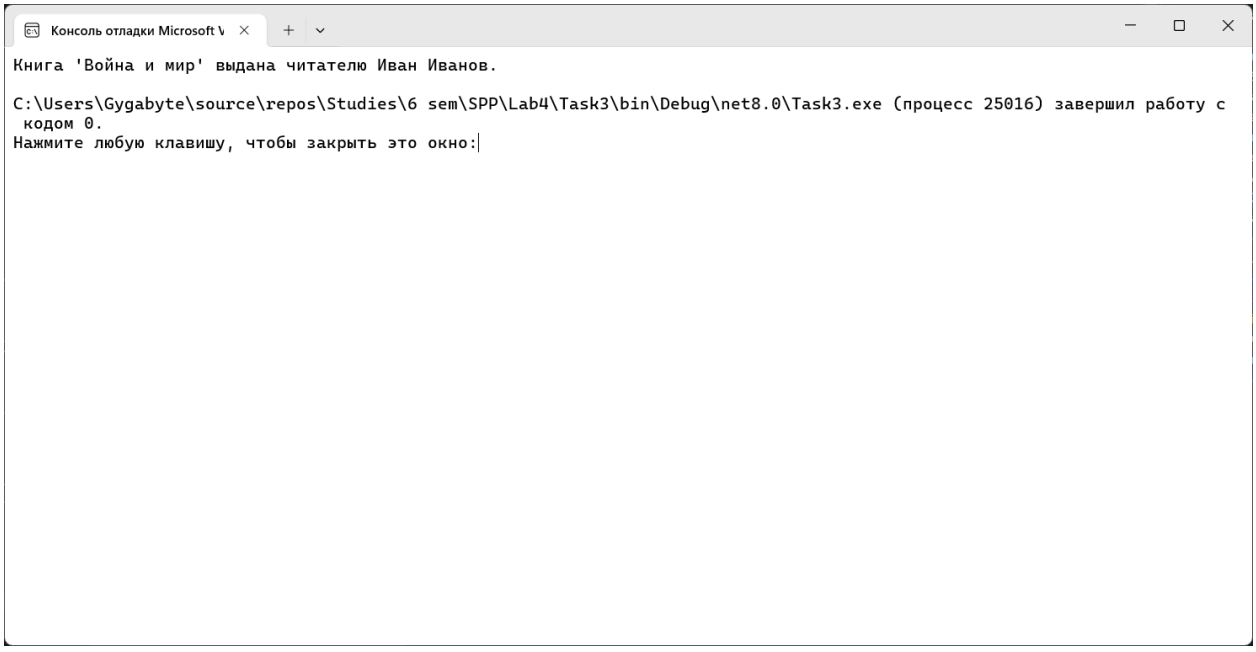
Система Библиотека. Читатель оформляет Заказ на Книгу. Система осуществляет поиск в Каталоге. Библиотекарь выдает Читателю Книгу на абонемент или в читальный зал. При невозвращении Книги Читателем он может быть занесен Администратором в «черный список».

```
public class Book
{
    public required string Title { get; set; }
    public required string Author { get; set; }
    public bool IsAvailable { get; set; }
}



public class Catalog
{
    private readonly List<Book> books = [];

    public void AddBook(Book book)
    {
        books.Add(book);
    }

    public Book? FindBook(string title)
    {
        return books.Find(b => b.Title.Equals(title,
StringComparison.OrdinalIgnoreCase) && b.IsAvailable);
    }
}



public class Reader
{
    public required string Name { get; set; }
    public bool IsBlacklisted { get; set; }
}
```

```
public class Order
{
    public required Book Book { get; set; }
    public required Reader Reader { get; set; }
}
```

```
public class Librarian
{
    public static Order? ProcessOrder(Reader reader, string bookTitle, Catalog
catalog)
    {
        Book? book = catalog.FindBook(bookTitle);
        if (book != null && !reader.IsBlacklisted)
        {
            book.IsAvailable = false;
            return new Order { Book = book, Reader = reader };
        }
        return null;
    }
}
```

```
public class Administrator
{
    public static void BlacklistReader(Reader reader)
    {
        reader.IsBlacklisted = true;
    }
}
```



Вывод: приобрел практические навыки в области объектно-ориентированного проектирования.