**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

***Факультет Информационных технологий***

***Кафедра Информатики и информационных технологий***

**направление подготовки**

**09.03.02 «Информационные системы и технологии»**

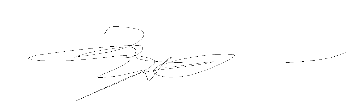
**ЛАБОРАТОРНАЯ РАБОТА № 2**

**Дисциплина:**

**Тема:** «Структуры. Типы значений и ссылочные типы»

**Выполнил: студент группы 241-334**

**Зотов Михаил Павлович**

**** (Фамилия И.О.)

25.09.2025

**Дата, подпись** **\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  ***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Проверил: \_\_\_\_\_\_\_\_*\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*** (Фамилия И.О., степень, звание) (Оценка)

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Замечания: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Москва**

**2025**

**Цель:** получить практические навыки в создании программ, содержащих добавление в программу структур. Изучить разницы в поведении типов значений и ссылочных типов. Изучить работу сборщика мусора, а также работы сборщика мусора с использованием пользовательских типов данных.

using System;

using System.Collections.Generic;

namespace LibraryManagementSystem

{

// Структура (тип значения) - информация о книге

public struct BookInfo

{

public string Title;

public string Author;

public int Year;

public decimal Price;

public BookInfo(string title, string author, int year, decimal price)

{

Title = title;

Author = author;

Year = year;

Price = price;

}

public void DisplayInfo()

{

Console.WriteLine($"Книга: {Title}");

Console.WriteLine($"Автор: {Author}");

Console.WriteLine($"Год: {Year}");

Console.WriteLine($"Цена: {Price:C}");

Console.WriteLine("---");

}

}

// Класс (ссылочный тип) - библиотека

public class Library

{

public string Name { get; set; }

public List<BookInfo> Books { get; set; }

public int BookCount => Books?.Count ?? 0;

public Library(string name)

{

Name = name;

Books = new List<BookInfo>();

}

public void AddBook(BookInfo book)

{

Books.Add(book);

Console.WriteLine($"Книга '{book.Title}' добавлена в библиотеку '{Name}'");

}

public void DisplayLibraryInfo()

{

Console.WriteLine($"\n=== Библиотека: {Name} ===");

Console.WriteLine($"Количество книг: {BookCount}");

foreach (var book in Books)

{

book.DisplayInfo();

}

}

// 4.1 Метод для создания копии класса с изменёнными параметрами

public Library CopyWithNewName(string newName)

{

var newLibrary = new Library(newName);

newLibrary.Books = new List<BookInfo>(this.Books); // Копирование списка

return newLibrary;

}

}

class Program

{

static void Main(string[] args)

{

Console.WriteLine("=== Лабораторная работа №2: Структуры и классы ===\n");

// Демонстрация различий между структурой и классом

DemonstrateValueVsReferenceTypes();

// Демонстрация работы с библиотекой

DemonstrateLibrarySystem();

// Демонстрация копирования

DemonstrateCopying();

Console.WriteLine("\nНажмите любую клавишу для выхода...");

Console.ReadKey();

}

static void DemonstrateValueVsReferenceTypes()

{

Console.WriteLine("1. ДЕМОНСТРАЦИЯ РАЗЛИЧИЙ МЕЖДУ СТРУКТУРОЙ И КЛАССОМ");

Console.WriteLine("===================================================");

// Работа со структурой (тип значения)

Console.WriteLine("\n--- Работа со структурой BookInfo ---");

BookInfo book1 = new BookInfo("Война и мир", "Лев Толстой", 1869, 1500m);

BookInfo book2 = book1; // Создается КОПИЯ структуры

Console.WriteLine("До изменения:");

Console.WriteLine($"book1.Title = {book1.Title}");

Console.WriteLine($"book2.Title = {book2.Title}");

// Изменяем book2

book2.Title = "Анна Каренина";

book2.Year = 1877;

Console.WriteLine("\nПосле изменения book2:");

Console.WriteLine($"book1.Title = {book1.Title} (не изменился)");

Console.WriteLine($"book2.Title = {book2.Title} (изменился)");

// Работа с классом (ссылочный тип)

Console.WriteLine("\n--- Работа с классом Library ---");

Library lib1 = new Library("Главная библиотека");

Library lib2 = lib1; // Ссылается на тот же объект

lib1.AddBook(book1);

Console.WriteLine($"lib1.BookCount = {lib1.BookCount}");

Console.WriteLine($"lib2.BookCount = {lib2.BookCount} (такой же, т.к. это ссылка)");

lib2.Name = "Новая библиотека";

Console.WriteLine($"lib1.Name = {lib1.Name} (изменился, т.к. это ссылка)");

}

static void DemonstrateLibrarySystem()

{

Console.WriteLine("\n\n2. ДЕМОНСТРАЦИЯ РАБОТЫ БИБЛИОТЕЧНОЙ СИСТЕМЫ");

Console.WriteLine("===========================================");

// Создание библиотеки

Library library = new Library("Центральная городская библиотека");

// Создание книг (структур)

BookInfo book1 = new BookInfo("Преступление и наказание", "Фёдор Достоевский", 1866, 1200m);

BookInfo book2 = new BookInfo("Мастер и Маргарита", "Михаил Булгаков", 1967, 1800m);

BookInfo book3 = new BookInfo("1984", "Джордж Оруэлл", 1949, 950m);

// Добавление книг в библиотеку

library.AddBook(book1);

library.AddBook(book2);

library.AddBook(book3);

// Отображение информации о библиотеке

library.DisplayLibraryInfo();

}

static void DemonstrateCopying()

{

Console.WriteLine("\n\n3. ДЕМОНСТРАЦИЯ КОПИРОВАНИЯ");

Console.WriteLine("===========================");

// Исходная книга

BookInfo originalBook = new BookInfo("Гарри Поттер", "Дж. К. Роулинг", 1997, 2000m);

Console.WriteLine("Исходная книга:");

originalBook.DisplayInfo();

// 4.2 Копирование структуры с изменением параметров (используя 'with')

BookInfo copiedBook = originalBook with { Title = "Гарри Поттер (копия)", Price = 1800m };

Console.WriteLine("Скопированная книга (с изменениями):");

copiedBook.DisplayInfo();

// Демонстрация, что это разные объекты

Console.WriteLine("Проверка, что это разные объекты:");

Console.WriteLine($"originalBook.Title = {originalBook.Title}");

Console.WriteLine($"copiedBook.Title = {copiedBook.Title}");

// Копирование класса

Library originalLib = new Library("Оригинальная библиотека");

originalLib.AddBook(originalBook);

Library copiedLib = originalLib.CopyWithNewName("Скопированная библиотека");

copiedLib.AddBook(copiedBook);

Console.WriteLine("\nОригинальная библиотека:");

originalLib.DisplayLibraryInfo();

Console.WriteLine("\nСкопированная библиотека:");

copiedLib.DisplayLibraryInfo();

// Демонстрация работы сборщика мусора

DemonstrateGarbageCollection();

}

static void DemonstrateGarbageCollection()

{

Console.WriteLine("\n\n4. ДЕМОНСТРАЦИЯ РАБОТЫ СБОРЩИКА МУСОРА");

Console.WriteLine("====================================");

// Создаем объекты, которые будут удалены сборщиком мусора

for (int i = 0; i < 5; i++)

{

Library tempLibrary = new Library($"Временная библиотека {i}");

BookInfo tempBook = new BookInfo($"Временная книга {i}", "Автор", 2000 + i, 100m \* i);

tempLibrary.AddBook(tempBook);

// Объекты выходят из области видимости и могут быть собраны сборщиком мусора

}

// Принудительный вызов сборщика мусора (в реальных приложениях обычно не требуется)

Console.WriteLine("Вызов сборщика мусора...");

GC.Collect();

GC.WaitForPendingFinalizers();

Console.WriteLine("Сборка мусора завершена. Временные объекты удалены.");

}

}

}