МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра интеллектуальных информационных технологий

Отчет по лабораторной работе №3

По дисциплине «Современные платформы программирования»

Специальность ПО-8

Выполнил:

Буртик Я.В.

студент группы ПО-8

Проверил:

ст. преп. кафедры ИИТ,

«\_\_»\_\_\_\_\_\_\_\_\_\_\_2024 г.

**Цель работы:** научиться создавать и использовать классы в программах на языке программирования С#

**Вариант 4**

**Задание 1.**

**Прямоугольник, заданный длинами двух сторон – Предусмотреть возможность определения площади и периметра, а так же логические методы, определяющие, является ли прямоугольник квадратом и существует ли такой прямоугольник. Конструктор должен позволить создавать объекты с начальной инициализацией. Реализовать метод equals, выполняющий сравнение объектов данного типа.**

Выполнение

**Код программы**

using sppLab3z1;  
  
Rectangle square = new Rectangle(3, 3);  
Rectangle rectangle = new Rectangle(10,4);  
Rectangle defaultRectangle = new Rectangle();  
Console.WriteLine(rectangle);  
Console.WriteLine(square.Perimeter());  
Console.WriteLine(rectangle.Square());  
Console.WriteLine(square.IsSquare());  
Console.WriteLine(rectangle.Equals(defaultRectangle));

namespace sppLab3z1;  
  
public class Rectangle  
{  
 private double \_a;  
 private double \_b;  
  
 public double A  
 {  
 get => \_a;  
 set => \_a = value;  
 }  
  
 public double B  
 {  
 get => \_b;  
 set => \_b = value;  
 }  
  
 public Rectangle(double a, double b)  
 {  
 \_a = a;  
 \_b = b;  
 }  
  
 public Rectangle()  
 {  
 \_a = 1d;  
 \_b = 1d;  
 }  
 public double Square()  
 {  
 return \_a \* \_b;  
 }  
  
 public double Perimeter()  
 {  
 return 2 \* (\_a + \_b);  
 }  
  
 public bool IsSquare()  
 {  
 if (\_a == \_b)  
 {  
 return true;  
 }  
 return false;  
 }  
  
 public override string ToString()  
 {  
 return $"a: {\_a}, b: {\_b}";  
 }  
  
 public bool Equals(Rectangle rectangle)  
 {  
 if (\_a == rectangle.\_a && \_b == rectangle.\_b)  
 {  
 return true;  
 }  
  
 return false;  
 }  
}

**Спецификация ввода**

Данные задаются программно  
Rectangle square = new Rectangle(3, 3);  
Rectangle rectangle = new Rectangle(10,4);  
Rectangle defaultRectangle = new Rectangle();

**Спецификация вывода**

a: длина rectangle, b: ширина rectangle

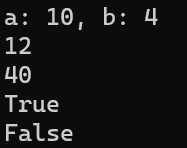
периметр square

площадь rectangle

квадрат ли square

равен ли rectangle и defaultRectangle

**Пример**



**Задание 2. Автоматизированная система в библиотеке**

**Составить программу, которая содержит текущую информацию о книгах в библиотеке. Сведе-**

**ния о книгах (Book) содержат:**

**• номер УДК;**

**• Фамилию и инициалы автора;**

**• Название;**

**• Год издания;**

**• Количество экземпляров в библиотеке;**

**• Количество страниц;**

**• Количество томов;**

**• ФИО читателя, взявшего книгу (при наличии);**

**4**

**Крощенко А.А., Современные платформы программирования, ЛР3, 2019**

**• Срок сдачи книги (если была взята).**

**Программа должна обеспечивать:**

**• Формирование общего списка книг;**

**• Формирование списка книг, старше n лет;**

**• Формирование списка книг, взятых на чтение;**

**• Формирование списка книг, взятых на чтение с выводом личной информации о читателях;**

**• Формирование списка книг, которые задержаны читателем дольше указанного срока.**

Выполнение

**Код программы**

using sppLab3z2;  
  
Book.ReadFromFile("books.txt");  
Console.WriteLine("All books:");  
Book.ShowAllBooks();  
Console.WriteLine("Show book older when 10 year:");  
Book.ShowBookOlder(10);  
Console.WriteLine("Show reader's books");  
Reader.ReadFromFile("readers.txt");  
Reader.ShowBook();  
Console.WriteLine("Show books and reader's data:");  
Reader.ShowBookAndReaderData();  
Console.WriteLine("Delay:");  
Reader.ShowDelayBook();

namespace sppLab3z2;  
  
public class Book  
{  
 private static int \_year = 2024;  
 private static List<Book> \_books = new List<Book>();  
 private int \_id;  
 private string \_authorName;  
 private int \_yearOfPublishing;  
 private string \_title;  
 private int \_numOfBooks;  
 private int \_numOfPage;  
 private int \_numOfVolume;  
  
 public Book(int id, string authorName, int yearOfPublishing, string title, int numOfPage, int numOfVolume, int numOfBooks)  
 {  
 \_id = id;  
 \_authorName = authorName;  
 \_yearOfPublishing = yearOfPublishing;  
 \_title = title;  
 \_numOfPage = numOfPage;  
 \_numOfVolume = numOfVolume;  
 \_numOfBooks = numOfBooks;  
 }  
  
 public static void ReadFromFile(string path)  
 {  
 string[] input;  
 using (StreamReader reader = new StreamReader(path))  
 {  
 input = reader.ReadToEnd().Split("\n");  
 }  
  
 foreach (var str in input)  
 {  
 string[] data = str.Split("~");  
 \_books.Add(new Book(  
 Convert.ToInt32(data[0]),  
 data[1],  
 Convert.ToInt32(data[2]),  
 data[3],  
 Convert.ToInt32(data[4]),  
 Convert.ToInt32(data[5]),  
 Convert.ToInt32(data[6])));  
 }  
 }  
   
 public static Book GetBook(int id)  
 {  
 foreach (var book in \_books)  
 {  
 if (book.\_id == id)  
 {  
 return book;  
 }  
 }  
  
 return null;  
 }  
  
 public static Book GetBook(string title)  
 {  
 foreach (var book in \_books)  
 {  
 if (book.\_title == title)  
 {  
 return book;  
 }  
 }  
  
 return null;  
 }  
  
 public static void ShowAllBooks()  
 {  
 foreach (var book in \_books)  
 {  
 Console.WriteLine(book);  
 }  
 }  
  
 public static void ShowBookOlder(int n)  
 {  
 foreach (var book in \_books)  
 {  
 if (\_year - book.\_yearOfPublishing > n)  
 {  
 Console.WriteLine(book);  
 }  
 }  
 }  
   
 public override string ToString()  
 {  
 return $"{\_id}|{\_title}|{\_authorName}|{\_yearOfPublishing}|{\_numOfPage}|{\_numOfVolume}|{\_numOfBooks}";  
 }  
}

namespace sppLab3z2;  
  
public class Reader  
{  
 private static List<Reader> \_readers = new List<Reader>();  
 private int \_id;  
 private string \_name;  
 private List<Book> \_books;  
 private List<DateTime> \_dateTimes;  
   
 public Reader(int id, string name)  
 {  
 \_id = id;  
 \_name = name;  
 \_books = new List<Book>();  
 \_dateTimes = new List<DateTime>();  
 }  
 public Reader(int id, string name, List<Book> books, List<DateTime> dateTimes)  
 {  
 \_id = id;  
 \_name = name;  
 \_books = books;  
 \_dateTimes = dateTimes;  
 }  
  
 public static void ReadFromFile(string path)  
 {  
 string[] input;  
 using (StreamReader reader = new StreamReader(path))  
 {  
 input = reader.ReadToEnd().Split("\n");  
 }  
  
 foreach (var str in input)  
 {  
 string[] data = str.Split("~");  
 int id = Convert.ToInt32(data[0]);  
 string name = data[1];  
 int numOfBook = Convert.ToInt32(data[2]);  
 List<int> booksId = new List<int>();  
 List<DateTime> dateTimes = new List<DateTime>();  
 for (int i = 0; i < numOfBook; i++)  
 {  
 booksId.Add(Convert.ToInt32(data[i+3]));   
 dateTimes.Add(Convert.ToDateTime(data[i+numOfBook+3]));  
 }  
 List<Book> books = new List<Book>();  
 foreach (var i in booksId)  
 {  
 books.Add(Book.GetBook(i));  
 }  
 \_readers.Add(new Reader(id,name,books,dateTimes));  
 }  
 }  
  
 public static void ShowBook()  
 {  
 foreach (var reader in \_readers)  
 {  
 foreach (var book in reader.\_books)   
 {  
 Console.WriteLine(book);  
 }  
 }  
 }  
   
 public static void ShowBookAndReaderData()  
 {  
 foreach (var reader in \_readers)  
 {  
 foreach (var book in reader.\_books)   
 {  
 Console.WriteLine($"{book}, Reader: {reader.\_id}|{reader.\_name}");  
 }  
 }  
 }  
 public static void ShowDelayBook()  
 {  
 foreach (var reader in \_readers)  
 {  
 int index = 0;  
 foreach (var book in reader.\_books)   
 {  
 if (reader.\_dateTimes[index] < DateTime.Now)  
 {  
 Console.WriteLine($"{book}, Reader: {reader.\_id}|{reader.\_name}");  
 }  
  
 index++;  
 }  
 }  
 }  
   
 public void AddBook(Book book, DateTime dateTime)  
 {  
 \_books.Add(book);  
 \_dateTimes.Add(dateTime);  
 }  
}

**Спецификация ввода**

**books.txt**

Id~name~year~title~numOfPagea~numOfVolume~numOfBooks

**Пример**

1~Author1~2020~Title1~100~1~40

2~Author2~2000~Title2~10~22~20

3~Author1~2010~Title3~140~1~106;Apple;123456789017;Tasty

**readers.txt**

Id~name~numOfBook~book1~…~datatime1~…

**Пример**

1~Reader1~3~1~3~2~20.10.2024~20.10.2024~20.10.2024

2~Reader2~2~3~2~20.10.2023~20.10.2024

3~Reader3~1~1~20.10.2023

**Спецификация вывода**

**Все книги**

**Все книги старше 10 лет**

**Все взятые книги**

**Все взятые книги и данные о читятелях**

**Все просроченные книги**

**Пример**

Изображение выглядит как текст, снимок экрана, Шрифт, черно-белый

Автоматически созданное описание

**Вывод**

Научились создавать и использовать классы в программах на языке программирования С#.