Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 1 з дисципліни
«Основи програмування 2. Модульне програмування»

«Файли даних. Бінарні файли»

Варіант 35

Виконав студент ІП-15, Шабанов Метін Шаміль огли
(шифр, прізвище, ім'я, по батькові)

Перевірив Вєчерковська Анастасія Сергіївна
(прізвище, ім'я, по батькові)

Київ 2022

Лабораторна робота 1.2

Файли даних. Бінарні файли

Варіант 35

Завдання:

35. Створити файл структур, що являє собою бібліотечний каталог і містить інформацію про книги, які є в наявності у бібліотеці: назва, автор (автори), рік видання, мова видання кількість екземплярів. Визначити кількість наявних екземплярів кожної книги вказаного автора. В новому файлі створити впорядкований за назвою каталог книг вказаного автора.

Виконання:

C#:

C# Main.cs | C# Book.cs | C# DataReader.cs | C# BinaryWorker.cs | C# AuthorsBooks.cs | C# SortBooks.cs | C# Printer.cs

```
using System.Collections.Generic;

namespace Lab2_BinaryFiles
{
    8 usages  8 inheritors  8 ext methods  8 exposing APIs
    class Program
    {
        8 usages  8 overrides  8 ext methods  8 exposing APIs
        static void Main(string[] args)
        {
            string inputFileName = "input.dat";
            string outputFileName = "output.dat";

            List<Book> catalogue = DataReader.Reader();
            BinaryWorker.BinWriter(inputFileName, catalogue);
            List<Book> readCatalogue = BinaryWorker.BinReader(inputFileName);

            Printer.PrintCatalogue(readCatalogue);
            List<Book> authorsBooks = AuthorsBooks.ReceiveAuthorsBooks(readCatalogue);
            Printer.PrintNumOfBooks(authorsBooks);

            authorsBooks = SortBooks.SortByName(authorsBooks);
            BinaryWorker.BinWriter(outputFileName, authorsBooks);
            Printer.PrintSortedBooks(authorsBooks);
        }
    }
}
```

C# Main.cs | C# Book.cs | C# DataReader.cs | C# BinaryWorker.cs | C# AuthorsBooks.cs | C# SortBooks.cs | C# Printer.cs

```
1
2       namespace Lab2_BinaryFiles
3       {
            34 usages
4           public struct Book
5           {
6               public string Name;
7               public string Author;
8               public int Year;
9               public string Language;
10              public int NumberOfCopies;
11          }
12      }
```

C# Main.cs | C# Book.cs | C# DataReader.cs | C# BinaryWorker.cs | C# AuthorsBooks.cs | C# SortBooks.cs | C# Printer.cs

```
1       using System;
2       using System.Collections.Generic;
3       using System.Linq;
4       using System.Text.RegularExpressions;
5
6       namespace Lab2_BinaryFiles
7       {
            1 usage
8           public class DataReader
9           {
                1 usage
10              public static List<Book> Reader()
11              {
12                  List<Book> structList = new List<Book>();
13
14                  Console.WriteLine("Press F1 to stop adding data.");
15                  while (Console.ReadKey().Key ≠ ConsoleKey.F1)
16                  {
17                      Book anotherBook = _dataCollector();
18                      structList = structList.Append(anotherBook).ToList();
19                      Console.WriteLine("Press F1 to stop adding data.");
20                  }
21
22                  return structList;
23              }
24
                1 usage
25              private static Book _dataCollector()
26              {
27                  Book anotherBook = new Book();
28                  string numPattern = @"\d";
29
```

C# Main.cs × C# Book.cs × C# DataReader.cs × C# BinaryWorker.cs × C# AuthorsBooks.cs × C# SortBooks.cs × C# Printer.cs ×

```
            Console.Write("Enter the name of the book: ");
            anotherBook.Name = Console.ReadLine();

            Console.Write("Enter the author: ");
            anotherBook.Author = Console.ReadLine();

            Console.Write("Enter the year of publication: ");
            anotherBook.Year = _testForPattern(Console.ReadLine(), numPattern);

            Console.Write("Enter the language: ");
            anotherBook.Language = Console.ReadLine();

            Console.Write("Enter the number of copies: ");
            anotherBook.NumberOfCopies =_testForPattern(Console.ReadLine(), numPattern);

            return anotherBook;
        }

        2 usages
        private static int _testForPattern(string testLine, string pattern)
        {
            while (!Regex.IsMatch( input: testLine, pattern))
            {
                Console.Write("This is not a number, try again: ");
                testLine = Console.ReadLine();
            }

            return int.Parse(testLine);
        }
    }
}
```

C# Main.cs × C# Book.cs × C# DataReader.cs × C# BinaryWorker.cs × C# AuthorsBooks.cs × C# SortBooks.cs × C# Printer.cs ×

```
using ...

namespace Lab2_BinaryFiles
{
    3 usages
    public class BinaryWorker
    {
        2 usages
        public static void BinWriter(string fileName, List<Book> structList)
        {
            FileSystem.FileOpen( FileNumber: 1, fileName, OpenMode.Random);

            foreach (Book book in structList)
            {
                FileSystem.FilePut( FileNumber: 1, book);
            }

            FileSystem.FileClose( params FileNumbers: 1);
        }

        1 usage
        public static List<Book> BinReader(string fileName)
        {
            FileSystem.FileOpen( FileNumber: 1, fileName, OpenMode.Random);

            List<Book> newStructList = new List<Book>();
            while (!(FileSystem.EOF( FileNumber: 1)))
            {
                ValueType tempBook = new Book();
                FileSystem.FileGet( FileNumber: 1, ref tempBook);
                newStructList = newStructList.Append((Book)tempBook).ToList();
            }
            FileSystem.FileClose( params FileNumbers: 1);

            return newStructList;
        }
    }
}
```

C# Main.cs C# Book.cs C# DataReader.cs C# BinaryWorker.cs C# AuthorsBooks.cs C# SortBooks.cs C# Printer.cs

```
{
    1 usage
    public class AuthorsBooks
    {
        1 usage
        public static List<Book> ReceiveAuthorsBooks(List<Book> catalogue)
        {
            Console.Write("Enter the author you`d like to find: ");
            string author = Console.ReadLine();

            int occurrences = 0;
            List<Book> authorsBooks = new List<Book>();
            authorsBooks = _findAuthor(catalogue, author, authorsBooks, ref occurrences);

            while (occurrences == 0)
            {
                Console.Write("There is no such author. Try again: ");
                author = Console.ReadLine();
                authorsBooks = _findAuthor(catalogue, author, authorsBooks, ref occurrences);
            }

            return authorsBooks;
        }

        2 usages
        private static List<Book> _findAuthor(List<Book> catalogue, string author,  List<Book> authorsBooks, ref int occurrences)
        {
            for (int i = 0; i < catalogue.Count; i++)
            {
                if (catalogue[i].Author == author)
                {
                    authorsBooks = authorsBooks.Append(catalogue[i]).ToList();
                    occurrences++;
                }
            }

            return authorsBooks;
        }
```

C# Main.cs C# Book.cs C# DataReader.cs C# BinaryWorker.cs C# AuthorsBooks.cs C# SortBooks.cs C# Printer.cs

```
        1 usage
        public static List<Book> SortByName(List<Book> authorsBooks)
        {
            for (int i = 0; i < authorsBooks.Count; i++)
            {
                int currentLetter = 0;
                for (int j = 0; j < authorsBooks.Count - i - 1;)
                {
                    authorsBooks = _swapByDemands(authorsBooks, ref j, ref currentLetter);
                }
            }

            return authorsBooks;
        }


        1 usage
        private static List<Book> _swapByDemands(List<Book> authorsBooks, ref int j, ref int currentLetter)
        {
            bool areSmallEnough = currentLetter < authorsBooks[j].Name.Length &&
                                  currentLetter < authorsBooks[j + 1].Name.Length;
            bool demands = authorsBooks[j].Name[currentLetter] == authorsBooks[j + 1].Name[currentLetter] &&
                           areSmallEnough;
            if (authorsBooks[j].Name[currentLetter] > authorsBooks[j + 1].Name[currentLetter])
            {
                (authorsBooks[j], authorsBooks[j + 1]) = (authorsBooks[j + 1], authorsBooks[j]);
                j++;
                currentLetter = 0;
            }
            else if (demands)
                currentLetter++;
            else
            {
                j++;
                currentLetter = 0;
            }

            return authorsBooks;
        }
```

```
Main.cs | Book.cs | DataReader.cs | BinaryWorker.cs | AuthorsBooks.cs | SortBooks.cs | Printer.cs

namespace Lab2_BinaryFiles
{
    3 usages
    public class Printer
    {
        1 usage
        public static void PrintCatalogue(List<Book> catalogue)
        {
            Console.WriteLine("\n\nFull catalogue:");
            foreach (Book book in catalogue)
            {
                Console.WriteLine();
                Console.WriteLine($"Name: {book.Name} | Author: {book.Author} | Year: {book.Year} | Language: {book.Language} | Number of copies: {book.NumberOfCopies}");
            }
        }

        1 usage
        public static void PrintNumOfBooks(List<Book> authorsBooks)
        {
            Console.WriteLine("\n\nNumber of copies for each book is:");
            foreach (Book book in authorsBooks)
            {
                Console.WriteLine();
                Console.WriteLine($"Author: {book.Author} | Name: {book.Name} | Number of copies: {book.NumberOfCopies}" );
            }
        }

        1 usage
        public static void PrintSortedBooks(List<Book> sortedBooks)
        {
            Console.WriteLine("\n\nSorted:");
            foreach (Book book in sortedBooks)
            {
                Console.WriteLine($"Author: {book.Author} | Name: {book.Name}");
            }
        }
    }
}
```

Python:

```
Main.py | BookFile.py | DataReader.py | BinaryWorker.py | AuthorsBooks.py | SortTheBooks.py | Printer.py

from BookFile import Book
import DataReader
import BinaryWorker
import AuthorsBooks
import Printer
import SortTheBooks

inputFileName = "input.dat"
outputFileName = "output.dat"

catalogue: list[Book] = DataReader.reader()
BinaryWorker.BinWriter(inputFileName, catalogue)
readCatalogue: list[Book] = BinaryWorker.BinReader(inputFileName)

Printer.printCatalogue(readCatalogue)
authorsBooks: list[Book] = AuthorsBooks.receiveAuthorsBooks(readCatalogue)
Printer.printNumOfBooks(authorsBooks)

sortedBooks = SortTheBooks.SortBooks(authorsBooks)
BinaryWorker.BinWriter(outputFileName, sortedBooks)
Printer.printSortedBooks(authorsBooks)
```

Main.py | BookFile.py | DataReader.py | BinaryWorker.py | AuthorsBooks.py | SortTheBooks.py | Printer.py

```python
class Book:
    name = 'Default Name'
    author = 'Default Author'
    year = None
    language = 'Default Language'
    numberOfCopies = None
```

Main.py | BookFile.py | DataReader.py | BinaryWorker.py | AuthorsBooks.py | SortTheBooks.py | Printer.py

```python
from BookFile import Book


def reader():
    structList = []

    while True:
        print('Press slash to stop adding data.')
        endReading = input()
        if endReading == "/":
            break
        else:
            newBook = dataCollector()
            structList.append(newBook)

    return structList


def dataCollector():
    anotherBook = Book()
    anotherBook.name = input('Enter the name of the book: ')
    anotherBook.author = input('Enter the author: ')

    anotherBook.year = input('Enter the year: ')
    while not anotherBook.year.isnumeric():
        anotherBook.year = input('That`s not a number, enter the valid year: ')

    anotherBook.language = input('Enter the language: ')
    anotherBook.numberOfCopies = input('Enter the number of copies: ')

    while not anotherBook.numberOfCopies.isnumeric():
        anotherBook.numberOfCopies = input('That`s not a number, enter valid data: ')

    return anotherBook
```

Main.py | BookFile.py | DataReader.py | BinaryWorker.py | AuthorsBooks.py | SortTheBooks.py | Printer.py

```python
import pickle
from BookFile import Book


def BinWriter(filename, booklist: list[Book]):
    file = open(filename, 'wb')
    pickle.dump(booklist, file)
    file.close()


def BinReader(filename):
    with open(filename, 'rb') as file:
        bookList: list[Book] = pickle.load(file)
    return bookList
```

Main.py | BookFile.py | DataReader.py | BinaryWorker.py | AuthorsBooks.py | SortTheBooks.py | Printer.py

```python
from BookFile import Book


def receiveAuthorsBooks(catalogue):
    author = input('Enter the author you`d like to find: ')
    occurrences = 0
    authorsBooks = []
    for books in catalogue:
        if author == books.author:
            authorsBooks.append(books)
            occurrences += 1

    while occurrences == 0:
        author = input('There is no such author. Try again:  ')
        for books in catalogue:
            if author == books.author:
                authorsBooks.append(books)
                occurrences += 1

    return authorsBooks
```

Main.py | BookFile.py | DataReader.py | BinaryWorker.py | AuthorsBooks.py | SortTheBooks.py | Printer.py

```python
from BookFile import Book


def SortBooks(authorsBooks: list[Book]):
    i = 0
    while i < len(authorsBooks):
        currentLetter = 0
        j = 0
        while j < len(authorsBooks) - i - 1:
            if authorsBooks[j].name[currentLetter] > authorsBooks[j + 1].name[currentLetter]:
                authorsBooks[j], authorsBooks[j + 1] = authorsBooks[j + 1], authorsBooks[j]
                j += 1
                currentLetter = 0
            elif authorsBooks[j].name[currentLetter] == authorsBooks[j + 1].name[currentLetter] and currentLetter < len(authorsBooks[j].name) and currentLetter < len(authorsBooks[j + 1].name):
                currentLetter += 1
            else:
                j += 1
                currentLetter = 0
        i += 1

    return authorsBooks
```

Main.py | BookFile.py | DataReader.py | BinaryWorker.py | AuthorsBooks.py | SortTheBooks.py | Printer.py

```python
from BookFile import Book


def printCatalogue(catalogue):
    print('\n\nFull catalogue:')
    for book in catalogue:
        line = f'Name: {book.name} | Author: {book.author} | Year: {book.year} '
        line += f'| Language: {book.language} | Number of copies: {book.numberOfCopies}'
        print(line)


def printNumOfBooks(authorsBooks):
    print('\n\nNumber of copies for each book is:')
    for book in authorsBooks:
        print(f'Author: {book.author} | Name: {book.name} | Number of copies: {book.numberOfCopies}')


def printSortedBooks(sortedBooks):
    print('\n\nSorted:')
    for book in sortedBooks:
        print(f'Author: {book.author} | Name: {book.name}')

```

Тестування:

C#:

```
"C:\Program Files\JetBrains\JetBrains Rider 2021.3.3\plugins\dpa\DotFiles\JetBrains.DPA.Runner.exe" --handle=104
Press F1 to stop adding data.
Enter the name of the book: Pure Code
Enter the author: R Martin
Enter the year of publication: 2000
Enter the language: English
Enter the number of copies: 10000
Press F1 to stop adding data.
Enter the name of the book: RR Hood
Enter the author: Unknown
Enter the year of publication: 0
Enter the language: German
Enter the number of copies: 1000900
Press F1 to stop adding data.
Enter the name of the book: Harry Potter
Enter the author: Rowling
Enter the year of publication: 1991
Enter the language: English
Enter the number of copies: 2000000
Press F1 to stop adding data.
Enter the name of the book: Saryw Potuh
Enter the author: Rowling
Enter the year of publication: 1966
Enter the language: Mongolian
Enter the number of copies: 3
Press F1 to stop adding data.
Enter the name of the book: Azry Hoto
Enter the author: Rowling
Enter the year of publication: 2005
Enter the language: French
Enter the number of copies: 0
Press F1 to stop adding data.


Full catalogue:

Name: Pure Code | Author: R Martin | Year: 2000 | Language: English | Number of copies: 10000

 Name: RR Hood | Author: Unknown | Year: 0 | Language: German | Number of copies: 1000900

 Name: Harry Potter | Author: Rowling | Year: 1991 | Language: English | Number of copies: 2000000

 Name: Saryw Potuh | Author: Rowling | Year: 1966 | Language: Mongolian | Number of copies: 3

 Name: Azry Hoto | Author: Rowling | Year: 2005 | Language: French | Number of copies: 0
 Enter the author you`d like to find: Rowling


 Number of copies for each book is:

 Author: Rowling | Name: Harry Potter | Number of copies: 2000000

 Author: Rowling | Name: Saryw Potuh | Number of copies: 3

 Author: Rowling | Name: Azry Hoto | Number of copies: 0


 Sorted:
 Author: Rowling | Name: Azry Hoto
 Author: Rowling | Name: Harry Potter
 Author: Rowling | Name: Saryw Potuh

 Process finished with exit code 0.
```

Python:

```
Press slash to stop adding data.

Enter the name of the book: 1984
Enter the author: Orwell
Enter the year: 1948
Enter the language: English
Enter the number of copies: 100000
Press slash to stop adding data.

Enter the name of the book: Harry Potter
Enter the author: Rowling
Enter the year: 1991
Enter the language: English
Enter the number of copies: 20000
Press slash to stop adding data.

Enter the name of the book: Aary Potter
Enter the author: Rowling
Enter the year: 2000
Enter the language: Egyptian
Enter the number of copies: 20
Press slash to stop adding data.

Enter the name of the book: Hazary Pot
Enter the author: Rowling
Enter the year: 1867
Enter the language: French
Enter the number of copies: 0
Press slash to stop adding data.

Enter the name of the book: Foundation
Enter the author: Asimov
Enter the year: 1950
Enter the language: English
Enter the number of copies: 20000000
Press slash to stop adding data.
Press slash to stop adding data.
/


Full catalogue:
Name: 1984 | Author: Orwell | Year: 1948 | Language: English | Number of copies: 100000
Name: Harry Potter | Author: Rowling | Year: 1991 | Language: English | Number of copies: 20000
Name: Aary Potter | Author: Rowling | Year: 2000 | Language: Egyptian | Number of copies: 20
Name: Hazary Pot | Author: Rowling | Year: 1867 | Language: French | Number of copies: 0
Name: Foundation | Author: Asimov | Year: 1950 | Language: English | Number of copies: 20000000
Enter the author you`d like to find: Rowling


Number of copies for each book is:
Author: Rowling | Name: Harry Potter | Number of copies: 20000
Author: Rowling | Name: Aary Potter | Number of copies: 20
Author: Rowling | Name: Hazary Pot | Number of copies: 0


Sorted:
Author: Rowling | Name: Aary Potter
Author: Rowling | Name: Harry Potter
Author: Rowling | Name: Hazary Pot

Process finished with exit code 0
```