

Лабораторная работа №3

«ИССЛЕДОВАНИЕ КОЛЛЕКЦИЙ И ИТЕРАТОРОВ В ЯЗЫКЕ JAVA»

3.1 Цель работы:

В ходе выполнения данной лабораторной работы необходимо ознакомиться с организацией коллекций объектов на языке Java, приобрести практические навыки использования списков, очередей, хеш-таблиц при создании Java программ.

3.2 Постановка задачи

Таблица 1 – Задание по варианту (Вариант 11)

Тип информации	Поле для сортировки (P)	Направление (U)	Тип коллекции (T1)	Тип коллекции (T2)
А	3	Убывание	TreeSet	TreeMap

А: Книга (Автор, Год издания, Количество страниц, Издательство).

Требуется реализовать класс А.

Реализовать коллекцию типа T1, объектов разработанного класса А. Реализовать возможность ввода объектов из файла, вывода на консоль, проверки присутствия объекта в коллекции по полю 1 класса А вводимого с клавиатуры пользователем. Имя файла вводить параметром командной строки -i.

Реализовать коллекцию типа LinkedList объектов класса А с возможностью упорядочивания по полю 1 (использовать Collections.sort(list)); с возможностью упорядочивания по полю P в направлении U класса (использовать Collections.sort(list, myComp), где myComp – экземпляр разработанного класса, реализующего интерфейс Comparator); с возможностью ввода элементов из файла, вывода на консоль и сохранения в файл. Имена файлов вводить параметрами командной строки -i и -o.

Реализовать коллекцию типа T2 объектов класса А с ключом по значению поля 1, с возможностью ввода элементов из файла, вывода на консоль в виде «Ключ ->

Значения», вывода значения полей по введенному с консоли значению поля 1. Имя файла вводить параметром командной строки `-i`.

В методе `main` реализовать следующие действия:

- Ввести записи из файла, заданного параметром командной строки `-i` в коллекцию `T1`.
- Отобразить записи в консоли.
- Предложить пользователю ввести значение поля 1.
- Отобразить в консоли результат проверки наличия записи по введенному значению поля 1.
- Ввести записи из файла, заданного параметром командной строки `-i` в коллекцию `LinkedList`.
- Отобразить записи в консоли. Отсортировать по полю 1. Отобразить записи в консоли. Отсортировать по полю `P` в направлении `U`. Отобразить записи в консоли.
- Вывести записи в файл, заданный параметром командной строки `-o`.
- Ввести записи из файла, заданного параметром командной строки `-i` в коллекцию `T2`.
- Отобразить записи в консоли.
- Предложить пользователю ввести значение поля 1.
- Отобразить в консоли значения остальных полей по введенному значению поля 1.

3.3 Ход работы

3.3.1 Текст программы

Была разработана программа на языке Java и представлена в листингах 1-6.

Листинг 1 – Класс `App` (содержит класс `main`)

```
import java.io.*;
import java.util.*;
import java.util.concurrent.TimeUnit;
```

```

public class App {

    public static void main (String[] args) {

        String inputFileName = "";
        String outputFileName = "";
        if(args.length > 0) {
            for (int i = 0; i < args.length; i++) {
                if (args[i].equals("-i")) {
                    inputFileName = args[++i];
                    System.out.println("Usage:  -i,  filename  -  <"  +
inputFileName + ">");
                }
                else {
                    inputFileName = "info.txt";
                }
                if (args[i].equals("-o")) {
                    outputFileName = args[++i];
                    System.out.println("Usage:  -o,  filename  -  <"  +
outputFileName + ">");
                }
                else {
                    outputFileName = "outputInfo.txt";
                }
            }
        }
        if (inputFileName.equals("")) {
            inputFileName = "info.txt";
        }
        if (outputFileName.equals("")) {
            outputFileName = "outputInfo.txt";
        }

        //-----

        // Ввести записи из файла заданного параметром командной строки -i в
коллекцию T1
        TaskOne T1 = new TaskOne();
        try {
            T1.readFromFile(inputFileName);
            System.out.println("Файл считан в коллекцию типа TreeSet");
        } catch (IOException e) {

```

```

        System.out.println("Ошибка при работе с файлом: " + e.getMessage());
    }

    //Отобразить записи в консоли
    T1.printBooks();

    //Предложить пользователю ввести значение поля 1 (autor)
    Scanner scanner = new Scanner(System.in);
    System.out.print("Введите автора книги: ");
    String author = scanner.nextLine();

    //Отобразить в консоли результат проверки наличия записи по введенному
    значению поля 1
    if (T1.containsBookByAuthor(author)) {
        System.out.println("Указанный Вами автор содержится в наших
записях");
    }
    else {
        System.out.println("Указанный Вами автор не содержится в наших
записях");
    }

    //-----
    System.out.println();

    //Ввести записи из файла, заданного параметром командной строки -i в
коллекцию LinkedList
    TaskTwo T2 = new TaskTwo();
    try {
        T2.readFromFile(inputFileName);
        System.out.println("Файл считан в коллекцию типа linkedList");
    } catch (IOException e) {
        System.out.println("Ошибка при работе с файлом: " + e.getMessage());
    }

    //Отобразить записи в консоли
    T2.printToConsole();

    //Отсортировать по полю 1 (autor)
    T2.sortByAuthor();
    System.out.println("*Сортировка по полю autor*");

```

```

//Отобразить записи в консоли
T2.printToConsole();

//Отсортировать по полю Р (pageCount) в направлении U (по убыванию)
T2.sortByPageCount();
System.out.println("*Сортировка по полю кол-во страниц (по убыванию)*");

//Отобразить записи в консоли
T2.printToConsole();

//Вывести записи в файл, заданный параметром командной строки -o
try {
    T2.saveToFile(outputFileName);
    System.out.println("Данные сохранены в файл");

} catch (IOException e) {
    System.out.println("Ошибка при работе с файлом: " + e.getMessage());
}

//-----
System.out.println();

//Ввести записи из файла, заданного параметром командной строки -i в
коллекцию T2
TaskThree T3 = new TaskThree();
try {
    T3.readFromFile(inputFileName);
    System.out.println("Файл считан в коллекцию типа TreeMap");
} catch (IOException e) {
    System.out.println("Ошибка при работе с файлом: " + e.getMessage());
}

//Отобразить записи в консоли
T3.printToConsole();

//Предложить пользователю ввести значение поля 1.
//Отобразить в консоли значения остальных полей по введенному значению поля
1.

T3.printBookByAuthor();

//-----

```

```

        try {
            TimeUnit.SECONDS.sleep(100); // заснуть на 100 секунд
        } catch (InterruptedException e) {
            System.out.println("Ошибка");
        }

        scanner.close();
        return;
    }
}

```

Листинг 2 – Класс Book

```

public class Book implements Comparable<Book> {
    private String author;
    private int year;
    private int pageCount;
    private String publisher;

    public Book(String author, int year, int pageCount, String publisher) {
        this.author = author;
        this.year = year;
        this.pageCount = pageCount;
        this.publisher = publisher;
    }

    // Метод для сравнения книг по автору
    @Override
    public int compareTo(Book other) {
        return this.author.compareTo(other.author);
    }

    // Метод для вывода информации о книге на консоль
    @Override
    public String toString() {
        return author + " " + year + " " + pageCount + " " + publisher;
    }

    // геттеры и сеттеры для всех полей

```

```

public String getAuthor() {
    return author;
}

public void setAuthor(String author) {
    this.author = author;
}

public int getYear() {
    return year;
}

public void setYear(int year) {
    this.year = year;
}

public int getPageCount() {
    return pageCount;
}

public void setPageCount(int pageCount) {
    this.pageCount = pageCount;
}

public String getPublisher() {
    return publisher;
}

public void setPublisher(String publisher) {
    this.publisher = publisher;
}
}

```

Листинг 3 – Класс TaskOne (для работы с коллекцией типа TreeSet)

```

import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;
import java.util.TreeSet;

public class TaskOne {

```

```

//единственное поле в классе - это коллекция
private TreeSet<Book> bookSet;

//конструктор
public TaskOne() {
    this.bookSet = new TreeSet<>();
}

//Метод для добавления книги в коллекцию
public void addBook(Book book) {
    bookSet.add(book);
}

// Метод для ввода книг из файла
public void readFromFile(String filename) throws FileNotFoundException {
    File file = new File(filename);
    Scanner scanner = new Scanner(file);

    while (scanner.hasNextLine()) {
        //данные из файла считываются построчно, данные об одной книге записаны в
        одну строку и разделены ";"
        String line = scanner.nextLine();
        String[] tokens = line.split("; ");

        // Создаем новый объект Book и добавляем его в коллекцию
        Book book = new Book(tokens[0], Integer.parseInt(tokens[1]),
        Integer.parseInt(tokens[2]), tokens[3]);
        addBook(book);
    }
    scanner.close();
}

// Метод для вывода всех книг в коллекции на консоль
public void printBooks() {
    System.out.println("-----");
    for (Book book : bookSet) {
        System.out.println(book);
    }
    System.out.println("-----
");
}

```



```

// Метод для проверки наличия книги по автору
public boolean containsBookByAuthor(String author) {
    for (Book book : bookSet) {
        if (book.getAuthor().equals(author)) {
            return true;
        }
    }
    return false;
}
}

```

Листинг 4 – Класс TaskTwo (для работы с коллекцией типа LinkedList)

```

import java.io.*;
import java.util.*;

public class TaskTwo {

    private LinkedList<Book> list;

    public TaskTwo() {
        list = new LinkedList<>();
    }

    public void addBook(Book book) {
        list.add(book);
    }

    public void sortByAuthor() {
        Collections.sort(list);
    }

    public void sortByPageCount() {
        Collections.sort(list, new BookComparator());
    }

    public void readFromFile(String filename) throws IOException {
        BufferedReader reader = new BufferedReader(new FileReader(filename));
        String line;
    }
}

```

```

while ((line = reader.readLine()) != null) {
    //метод split разделяет строку на элементы через разделитель
    //метод trim обрезает пробелы до и после строки
    String[] data = line.split("; ");
    String author = data[0].trim();
    int year = Integer.parseInt(data[1].trim());
    int pageCount = Integer.parseInt(data[2].trim());
    String publisher = data[3].trim();
    addBook(new Book(author, year, pageCount, publisher));
}
reader.close();
}

public void printToConsole() {
    System.out.println("-----");
    for (Book book : list) {
        System.out.println(book);
    }
    System.out.println("-----");
};
}

public void saveToFile(String filename) throws IOException {
    BufferedWriter writer = new BufferedWriter(new FileWriter(filename));
    for (Book book : list) {
        writer.write(book.getAuthor() + "; " + book.getYear() + "; " +
book.getPageCount() + "; " + book.getPublisher());
        writer.newLine();
    }
    writer.close();
}
}

```

Листинг 5 – Класс BookComparator

```

import java.util.Comparator;

class BookComparator implements Comparator<Book> {
    @Override
    public int compare(Book b1, Book b2) {
        return Integer.compare(b2.getPageCount(), b1.getPageCount());
    }
}

```

Листинг 7 – TaskThree (для работы с коллекцией типа TreeMap)

```
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Map;
import java.util.Scanner;
import java.util.TreeMap;

//Реализовать коллекцию типа T2 (TreeMap) объектов класса A (Book) с ключом по значению
поля 1 (autor),
//с возможностью ввода элементов из файла, вывода на консоль в виде «Ключ -> Значения»,
//вывода значения полей по введенному с консоли значению поля 1 (autor).
//Имя файла вводить параметром командной строки -i.

public class TaskThree {

    private TreeMap<String, Book> booksByAuthor;

    public TaskThree () {
        booksByAuthor = new TreeMap<>();
    }

    public void readFromFile(String fileName) throws FileNotFoundException {
        File file = new File(fileName);
        Scanner scanner = new Scanner(file);
        while (scanner.hasNextLine()) {
            String line = scanner.nextLine();
            String[] parts = line.split("; ");
            String author = parts[0];
            int year = Integer.parseInt(parts[1]);
            int pageCount = Integer.parseInt(parts[2]);
            String publisher = parts[3];
            Book book = new Book(author, year, pageCount, publisher);
            booksByAuthor.put(author, book);
        }
        scanner.close();
    }

    public void printToConsole() {
        System.out.println("-----");
        for (Map.Entry<String, Book> entry : booksByAuthor.entrySet()) {
```

```

        String author = entry.getKey();
        Book book = entry.getValue();

        System.out.println(author + " -> " + book.toString());
    }
    System.out.println("-----");
");
    }

    public void printBookByAuthor() {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Введите имя автора: ");
        String author = scanner.nextLine();
        Book book = booksByAuthor.get(author);
        if (book == null) {
            System.out.println("Книга не найдена");
        } else {
            System.out.println(book.toString());
        }
        scanner.close();
    }
}

```

3.3.2 Тестирование программы

Программа была скомпилирована в консоли и запущена. В программу были переданы имена файлов со входными и выходными данными в качестве параметров командной строки. Результат запуска представлен на рисунке 1. Входные данные были получены из текстового файла (рисунок 2). Программа вывела на экран корректные, ожидаемые данные. В результате работы программы был создан текстовый файл, содержащий выходные данные (рисунок 3).

Результаты тестирования полностью соответствуют ожиданиям.

```

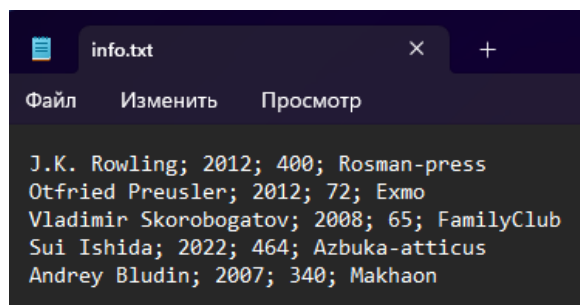
PS D:\5_semester\Java\Lab_3\program attempt 2\App\src> javac *.java
PS D:\5_semester\Java\Lab_3\program attempt 2\App\src> java -cp . App -i info.txt -o outputInfo.txt
Usage: -i, filename - <info.txt>
Usage: -o, filename - <outputInfo.txt>
Файл считан в коллекцию типа TreeSet
-----
Andrey Bludin 2007 340 Makhaon
J.K. Rowling 2012 400 Rosman-press
Otfried Preusler 2012 72 Exmo
Sui Ishida 2022 464 Azbuka-atticus
Vladimir Skorobogatov 2008 65 FamilyClub
-----
Введите автора книги: Sui Ishida
Указанный Вами автор содержится в наших записях

Файл считан в коллекцию типа linkedList
-----
J.K. Rowling 2012 400 Rosman-press
Otfried Preusler 2012 72 Exmo
Vladimir Skorobogatov 2008 65 FamilyClub
Sui Ishida 2022 464 Azbuka-atticus
Andrey Bludin 2007 340 Makhaon
-----
*Сортировка по полю autor*
-----
Andrey Bludin 2007 340 Makhaon
J.K. Rowling 2012 400 Rosman-press
Otfried Preusler 2012 72 Exmo
Sui Ishida 2022 464 Azbuka-atticus
Vladimir Skorobogatov 2008 65 FamilyClub
-----
*Сортировка по полю кол-во страниц (по убыванию)*
-----
Sui Ishida 2022 464 Azbuka-atticus
J.K. Rowling 2012 400 Rosman-press
Andrey Bludin 2007 340 Makhaon
Otfried Preusler 2012 72 Exmo
Vladimir Skorobogatov 2008 65 FamilyClub
-----
Данные сохранены в файл

Файл считан в коллекцию типа TreeMap
-----
Andrey Bludin -> Andrey Bludin 2007 340 Makhaon
J.K. Rowling -> J.K. Rowling 2012 400 Rosman-press
Otfried Preusler -> Otfried Preusler 2012 72 Exmo
Sui Ishida -> Sui Ishida 2022 464 Azbuka-atticus
Vladimir Skorobogatov -> Vladimir Skorobogatov 2008 65 FamilyClub
-----
Введите имя автора: Andrey Bludin
Andrey Bludin 2007 340 Makhaon
|

```

Рисунок 1 – Результат работы программы в консоли вывода



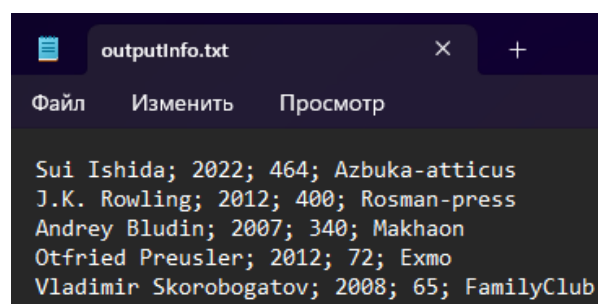
```

info.txt
Файл  Изменить  Просмотр

J.K. Rowling; 2012; 400; Rosman-press
Otfried Preusler; 2012; 72; Exmo
Vladimir Skorobogatov; 2008; 65; FamilyClub
Sui Ishida; 2022; 464; Azbuka-atticus
Andrey Bludin; 2007; 340; Makhaon

```

Рисунок 2 – Текстовый файл со входными данными



```

outputInfo.txt
Файл  Изменить  Просмотр

Sui Ishida; 2022; 464; Azbuka-atticus
J.K. Rowling; 2012; 400; Rosman-press
Andrey Bludin; 2007; 340; Makhaon
Otfried Preusler; 2012; 72; Exmo
Vladimir Skorobogatov; 2008; 65; FamilyClub

```

Рисунок 3 – Текстовый файл с выходными данными

Выводы

В ходе выполнения данной лабораторной работы ознакомились с организацией коллекций объектов на языке Java. Были приобретены практические навыки использования списков, очередей, хеш-таблиц при создании Java программ.

В результате работы было установлено, что каждая коллекция имеет свои преимущества и недостатки, которые необходимо учитывать при выборе подходящей коллекции для конкретной задачи. Например, ArrayList и LinkedList являются хорошими выборами для операций чтения и добавления элементов в конец коллекции, но при необходимости вставки или удаления элементов в середине коллекции рекомендуется использовать LinkedList.

Также было выяснено, что итераторы являются удобным и эффективным способом перебора элементов коллекции. Итераторы позволяют выполнять различные операции над коллекцией, такие как чтение, добавление, удаление элементов.

Полученные навыки и опыт помогут при дальнейшем программировании на языке Java.