МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«Севастопольский государственный университет»

кафедра Информационные системы

Сирота Марина Романовна

Институт информационных технологий и управления в технических системах

курс 3 группа ИС/б-32-о

09.03.02 Информационные системы и технологии (уровень бакалавриата)

ОТЧЕТ

по лабораторной работе №3

по дисциплине: «Платформа Java»

по теме: «Основы объектно-ориентированного программирования Java»

Отметка о зачете \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_

(дата)

Руководитель практикума

ст. пр. Кузнецов С.А.

(должность) (подпись) (инициалы, фамилия)

Севастополь

2018

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.

1. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

2.1. В соответствии с вариантом задания (см. таблицу 4.1) реализовать класс для представления требуемой информации.

2.2. Реализовать коллекцию типа Т1(см. таблицу 4.1) объектов разработанного в п. 2.1. класса с возможностью ввода элементов из файла, вывода на консоль, проверки членства по введенному с консоли значению поля 1. Имя файла вводить параметром командной строки –i.

2.3. Реализовать коллекцию типа LinkedList объектов разработанного в п. 2.1. с возможностью: упорядочивания по полю 1 (использовать Collections.sort(list)); с возможностью упорядочивания по полю P (см. таблицу 4.1) 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.

2.4. Реализовать коллекцию типа Т2(см. таблицу 4.1) объектов разработанного в п. 2.1. класса c ключом по значению поля 1, с возможностью ввода элементов из файла, вывода на консоль в виде «Ключ -> Значения» (значения остальных полей), вывода значения полей по введенному с консоли значению поля 1. Имя файла вводить параметром командной строки –i.

2.5. Реализовать класс Lab3Java, в методе main которого реализовать работу с объектами классов из п. 2.1-2.4:

1. Ввести записи из файла, заданного параметром командной строки –i в коллекцию T1.

2. Отобразить записи в консоли.

3. Предложить пользователю ввести значение поля 1.

4. Отобразить в консоли результат проверки наличия записи по введенному значению поля 1.

5. Ввести записи из файла, заданного параметром командной строки –i в коллекцию LinkedList.

6. Отобразить записи в консоли. Отсортировать по полю 1. Отобразить записи в консоли. Отсортировать по полю P в направлении U. Отобразить записи в консоли.7. Вывести записи в файл, заданный параметром командной строки –o.

8. Ввести записи из файла, заданного параметром командной строки –i в коллекцию T2.

9. Отобразить записи в консоли.

10. Предложить пользователю ввести значение поля 1.

11. Отобразить в консоли значения остальных полей по введенному значению поля 1.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2 | B | 2 | Возрастание | TreeSet | TreeMap |

B: Автомобиль (Марка, Год выпуска, Объем двигателя, Максимальная скорость).

1. ВЫПОЛНЕНИЕ РАБОТЫ
   1. Листинг

car.java

package jv3;

import java.util.Scanner;

public class car implements Comparable<car> {

protected final String marka;

protected final int year;

protected final float capacity;

protected final int maxSpeed;

public car(String marka, int year, float capacity, int maxSpeed) {

this.marka = marka;

this.year = year;

this.capacity = capacity;

this.maxSpeed = maxSpeed;

}

public String getMarka() {

return marka;

}

public int getYear() {

return year;

}

public float getPageCount() {

return capacity;

}

public int getPublisher() {

return maxSpeed;

}

public static car read(Scanner scn) {

return new car(scn.next(), scn.nextInt(), scn.nextFloat(), scn.nextInt());

}

@Override

public String toString() {

StringBuilder sb = new StringBuilder();

sb.append("car marka: ");

sb.append(marka);

sb.append("; year: ");

sb.append(year);

sb.append("; pages: ");

sb.append(capacity);

sb.append("; maxSpeed: ");

sb.append(maxSpeed);

sb.append(";");

return sb.toString();

}

public String toKeyValueString() {

StringBuilder sb = new StringBuilder();

sb.append(marka);

sb.append(" -> ");

sb.append(year);

sb.append("; ");

sb.append(capacity);

sb.append("; ");

sb.append(maxSpeed);

sb.append(";");

return sb.toString();

}

public String toRawString() {

StringBuilder sb = new StringBuilder();

sb.append(marka);

sb.append(" ");

sb.append(year);

sb.append(" ");

sb.append(capacity);

sb.append(" ");

sb.append(maxSpeed);

sb.append(" ");

return sb.toString();

}

@Override

public int compareTo(car o) {

return marka.compareTo(o.marka);

}

}

Main.java

package jv3;

import java.io.File;

import java.util.Collections;

import java.util.Scanner;

public class Main {

protected static String inFile = "";

protected static String outFile = "";

public static void main(String[] args) {

if (!init(args)) {

return;

}

Scanner scn = new Scanner(System.in);

listOne list1 = new listOne();

list1.read(inFile);

System.out.print("\tFirst list:\n");

list1.print();

System.out.print("Input book author to be found\n");

if (list1.contains(scn.next())) {

System.out.print("Book found\n");

} else {

System.out.print("No such book\n");

}

System.out.print("\n");

listTwo list2 = new listTwo();

list2.read(inFile);

System.out.print("\tSecond list unsorted:\n");

list2.print();

System.out.print("\tSecond list sorted by 1st field:\n");

Collections.sort(list2);

list2.print();

System.out.print("\tSecond list sorted by 2nd field:\n");

Collections.sort(list2, new carComparator());

list2.print();

list2.write(outFile);

System.out.print("\n");

listThree list3 = new listThree();

list3.read(inFile);

System.out.print("\tThird list:\n");

list3.print();

System.out.print("Input book author to be found\n");

list3.print(scn.next());

System.out.print("\n");

scn.close();

}

private static boolean init(String[] args) {

int i = 0;

while (i < args.length) {

if (args[i].equalsIgnoreCase("-i")) {

inFile = args[++i];

} else if (args[i].equalsIgnoreCase("-o")) {

outFile = args[++i];

} else {

System.out.printf("Unknown argument: ", args[i]);

}

i++;

}

if (inFile.isEmpty()) {

System.out.print("Input file is unset, use -i argument");

return false;

} else if (!(new File(inFile)).exists()) {

System.out.print("Input file does not exists");

return false;

}

if (outFile.isEmpty()) {

System.out.print("Output file is unset, use -o argument");

return false;

}

return true;

}

}

carComparator.java

package jv3;

import java.util.Comparator;

public class carComparator implements Comparator<car> {

@Override

public int compare(car c1, car c2) {

return c1.getYear() - c2.getYear();

}

}

listOne.java

package jv3;

import java.util.TreeSet;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.util.Scanner;

public class listOne extends TreeSet<car> {

public void read(String filename){

this.clear();

try(Scanner scn =new Scanner(new FileInputStream(filename))){

while(scn.hasNext()){

add(car.read(scn));

}

}catch(FileNotFoundException e){

System.out.println("File not found");

e.printStackTrace();

}

}

public void print(){

for(car c :this){

System.out.println(c.toString());

}

}

public boolean contains(String marka){

for(car c :this){

if(c.getMarka().equalsIgnoreCase(marka)){

return true;

}

}

return false;

}

}

listTwo.java

package jv3;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.FileOutputStream;

import java.io.PrintWriter;

import java.util.LinkedList;

import java.util.Scanner;

public class listTwo extends LinkedList<car> {

public void read(String filename){

this.clear();

try(Scanner scn =new Scanner(new FileInputStream(filename))){

while(scn.hasNext()){

add(car.read(scn));

}

}catch(FileNotFoundException e){

System.out.println("File not found");

e.printStackTrace();

}

}

public void print(){

for(car c :this){

System.out.println(c.toString());

}

}

public void write(String filename){

try(PrintWriter wrt =new PrintWriter(new FileOutputStream(filename))){

for(car c :this){

wrt.write(c.toRawString());

wrt.write('\n');

}

}catch(FileNotFoundException e){

System.out.println("File not found");

e.printStackTrace();

}

}

}

listThree.java

package jv3;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.util.LinkedHashMap;

import java.util.Scanner;

import java.util.TreeMap;

public class listThree extends TreeMap<String, car> {

public void read(String filename){

this.clear();

car c;

try(Scanner scn =new Scanner(new FileInputStream(filename))){

while(scn.hasNext()){

c = car.read(scn);

put(c.getMarka(),c);

}

}catch(FileNotFoundException e){

System.out.println("File not found");

e.printStackTrace();

}

}

public void print(){

for(car c :this.values()){

System.out.println(c.toKeyValueString());

}

}

public void print(String marka){

car c = get(marka);

if(c !=null){

System.out.println(c.toKeyValueString());

}else{

System.out.println("No such book");

}

}

}

* 1. Результаты выполнения

Рисунок 3.1 – Результаты в консоли

Рисунок 3.2 – Результаты в консоли

Рисунок 3.3 – Результаты в файлах

ВЫВОДЫ

В ходе данной лабораторной работы были изучены особенности организации коллекций объектов на языке Java.

Были приобретены практические навыки использования списков очередей, хеш-таблиц при создании Java программ.

Была написана программа в соответствии с вариантом задания.