МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М.Є. Жуковського “ХАІ”

Кафедра 603

Лабораторна робота № 5

**Робота з системою введення-виведення**

**у програмах Java**

з дисципліни “Програмування Інтернет”

ХАІ.603.631п.15В.050103.126337.ПЗ

Виконав студент гр.631п  Кошиль В. А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (№ групи) (П.І.Б)

(підпис, дата)

Перевірив:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(наукова ступінь, вчене звання)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис, дата) (П.І.Б)

2015

Цель работы: получение практических навыков работы с входными и выходными потоками данных в программах на языке Java.

Изучаемый материал: система ввода-вывода Java: основные концепции. Пакет **java.io**: основные классы, интерфейсы и методы. Особенности применения классов байтовых и символьных потоков ввода-вывода в приложениях. Классы исключений пакета **java.io**.

**Постановка задачи**

Разработать и реализовать на языке Java приложение для демонстрации работы с потоками ввода-вывода. В разрабатываемом приложении показать применение классов и методов пакета **java.io** для работы с различными источниками и приемниками данных (клавиатура, файловая система, память), а также возможности сохранения и восстановления состояния объектов с помощью потоков ввода-вывода.

Примечание. В качестве основы можно использовать программные приложения, разработанные в предыдущих лабораторных работах.

**Ход работы**

1. В качестве основы было взято приложение из лабораторной работы №2 – Автопарк. Исходная программа дополнена необходимым кодом, который поддерживает работу с системой ввода-вывода.

Main.java

…

import java.io.IOException;

. . .

String path = "D:/SkyDrive/3 курс/!2 cemester/!COMPLETED/Java/Laba5/collection.txt";

ReadWrite readerWriter = **null**;

. . .

readerWriter = **new** ReadWrite(path);

readerWriter.SaveCollection(transports);

. . .

readerWriter = **new** ReadWrite(path);

Vector<String> info = readerWriter.LoadCollection();

ReadWrite.java

**import** java.io.\*;

**import** java.util.Vector;

**public** **class** ReadWrite {

**private** String path;

**public** ReadWrite(String path){

**this**.path = path;

}

**public** **void** SaveCollection(Vector<Transport> collection){

Writer writer = **null**;

**try** {

writer = **new** FileWriter(**this**.path);

**for** (**int** i = 0; i<collection.size(); i++) {

writer.write("The "+(i+1)+" car in autopark: ");

writer.write("\n");

writer.write("Car made by: " + collection.elementAt(i).GetFirm());

writer.write("\n");

writer.write("Car type is: " + collection.elementAt(i).GetType());

writer.write("\n");

writer.write("Car number is: " + collection.elementAt(i).GetNumber());

writer.write("\n");

writer.write("Car course is: " + collection.elementAt(i).GetWay());

writer.write("\n");

writer.write("Car capacity is: " + collection.elementAt(i).GetCapacity());

writer.write("\n");

writer.write("Driver name is: " + collection.elementAt(i).GetDriver());

writer.write("\n");

}

writer.flush();

}

**catch** (IOException e) {

System.***err***.println(e);

}

**finally** {

**if** (writer != **null**) {

**try** {

writer.close();

}

**catch** (IOException ex) {

}

}

}

}

**public** Vector<String> LoadCollection() **throws** FileNotFoundException, IOException{

String line = **null**;

Vector<String> lines = **new** Vector<String>();

File file = **new** File(**this**.path);

BufferedReader br = **new** BufferedReader(**new** InputStreamReader(**new** FileInputStream(file)));

**while** ((line = br.readLine()) != **null**) {

lines.add(line);

}

br.close();

**return** lines;

}

}

1. Приложение откомпилировано и протестировано.

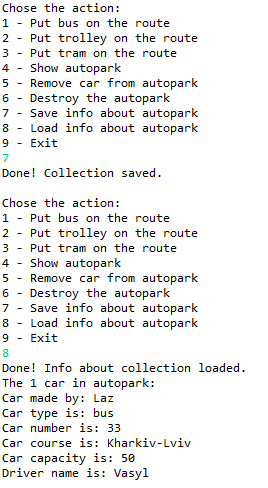
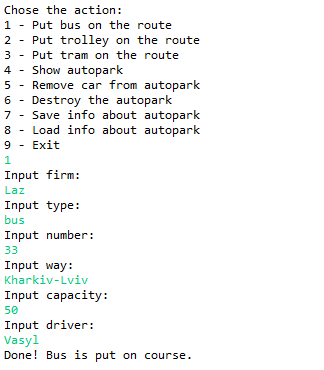


Рисунок 1 – Тестирование, работа с файлами

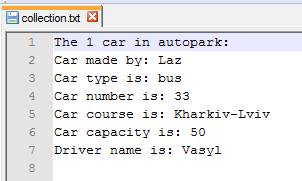


Рисунок 2 – Содержимое файла

**Выводы**: во время выполнения лабораторной работы были изучены теоретические сведения на тему систем ввода-вывода в языке Java. В качестве основы была взята программа из лабораторной работы №2. Исходный код дополнен всеми необходимыми компонентами и успешно протестирован. Также была получена документация Javadoc.

**Приложение А. Исходный код программы**

Main.java

**import** java.io.IOException;

**import** java.util.Scanner;

**import** java.util.Vector;

**public** **class** MainClass {

**private** **static** Scanner *scan*;

**public** **static** **void** main(String[] args) {

Vector<Transport> transports = **new** Vector<Transport>();

Transport transport;

**int** choose=0;

String path = "D:/SkyDrive/3 курс/!2 cemester/!COMPLETED/Java/Laba5/collection.txt";

ReadWrite readerWriter = **null**;

*scan* = **new** Scanner(System.***in***);

**while**(**true**){

System.***out***.println("\nChose the action:");

System.***out***.println("1 - Put bus on the route");

System.***out***.println("2 - Put trolley on the route");

System.***out***.println("3 - Put tram on the route");

System.***out***.println("4 - Show autopark");

System.***out***.println("5 - Remove car from autopark");

System.***out***.println("6 - Destroy the autopark");

System.***out***.println("7 - Save info about autopark");

System.***out***.println("8 - Load info about autopark");

System.***out***.println("9 - Exit");

choose = *scan*.nextInt();

**switch**(choose){

**case** 1:

transport = **new** Bus();

System.***out***.println("Input firm: ");

transport.SetFirm(*scan*.next());

System.***out***.println("Input type: ");

transport.SetType(*scan*.next());

System.***out***.println("Input number: ");

transport.SetNumber(*scan*.nextInt());

System.***out***.println("Input way: ");

transport.SetWay(*scan*.next());

System.***out***.println("Input capacity: ");

transport.SetCapacity(*scan*.nextInt());

System.***out***.println("Input driver: ");

transport.SetDriver(*scan*.next());

transports.add(transport);

System.***out***.println("Done! Bus is put on course.");

**break**;

**case** 2:

transport = **new** Trolley();

System.***out***.println("Input firm: ");

transport.SetFirm(*scan*.next());

System.***out***.println("Input type: ");

transport.SetType(*scan*.next());

System.***out***.println("Input number: ");

transport.SetNumber(*scan*.nextInt());

System.***out***.println("Input way: ");

transport.SetWay(*scan*.next());

System.***out***.println("Input capacity: ");

transport.SetCapacity(*scan*.nextInt());

System.***out***.println("Input driver: ");

transport.SetDriver(*scan*.next());

transports.add(transport);

System.***out***.println("Done! Trolley is put on course.");

**break**;

**case** 3:

transport = **new** Tram();

System.***out***.println("Input firm: ");

transport.SetFirm(*scan*.next());

System.***out***.println("Input type: ");

transport.SetType(*scan*.next());

System.***out***.println("Input number: ");

transport.SetNumber(*scan*.nextInt());

System.***out***.println("Input way: ");

transport.SetWay(*scan*.next());

System.***out***.println("Input capacity: ");

transport.SetCapacity(*scan*.nextInt());

System.***out***.println("Input driver: ");

transport.SetDriver(*scan*.next());

transports.add(transport);

System.***out***.println("Done! Tram is put on course.");

**break**;

**case** 4:

**for**(**int** i=0; i<transports.size(); i++){

System.***out***.println("The "+(i+1)+" car in autopark: ");

System.***out***.println("Car made by: " + transports.elementAt(i).GetFirm());

System.***out***.println("Car type is: " + transports.elementAt(i).GetType());

System.***out***.println("Car number is: " + transports.elementAt(i).GetNumber());

System.***out***.println("Car course is: " + transports.elementAt(i).GetWay());

System.***out***.println("Car capacity is: " + transports.elementAt(i).GetCapacity());

System.***out***.println("Driver name is: " + transports.elementAt(i).GetDriver());

}

**break**;

**case** 5:

System.***out***.println("Input position will be removed!");

transports.remove(*scan*.nextInt() - 1);

**break**;

**case** 6:

System.***out***.println("Autopark will be destroyed!");

transports.clear();

**break**;

**case** 7:

readerWriter = **new** ReadWrite(path);

readerWriter.SaveCollection(transports);

System.***out***.println("Done! Collection saved.");

**break**;

**case** 8:

readerWriter = **new** ReadWrite(path);

**try** {

Vector<String> info = readerWriter.LoadCollection();

System.***out***.println("Done! Info about collection loaded.");

**for**(String str : info){

System.***out***.println(str);

}

}

**catch** (IOException e) {

System.***err***.println(e);

}

**break**;

**case** 9:

System.*exit*(0);

}

}

}

}

ReadWrite.java

**import** java.io.\*;

**import** java.util.Vector;

**public** **class** ReadWrite {

**private** String path;

**public** ReadWrite(String path){

**this**.path = path;

}

**public** **void** SaveCollection(Vector<Transport> collection){

Writer writer = **null**;

**try** {

writer = **new** FileWriter(**this**.path);

**for** (**int** i = 0; i<collection.size(); i++) {

writer.write("The "+(i+1)+" car in autopark: ");

writer.write("\n");

writer.write("Car made by: " + collection.elementAt(i).GetFirm());

writer.write("\n");

writer.write("Car type is: " + collection.elementAt(i).GetType());

writer.write("\n");

writer.write("Car number is: " + collection.elementAt(i).GetNumber());

writer.write("\n");

writer.write("Car course is: " + collection.elementAt(i).GetWay());

writer.write("\n");

writer.write("Car capacity is: " + collection.elementAt(i).GetCapacity());

writer.write("\n");

writer.write("Driver name is: " + collection.elementAt(i).GetDriver());

writer.write("\n");

}

writer.flush();

}

**catch** (IOException e) {

System.***err***.println(e);

}

**finally** {

**if** (writer != **null**) {

**try** {

writer.close();

}

**catch** (IOException ex) {

}

}

}

}

**public** Vector<String> LoadCollection() **throws** FileNotFoundException, IOException{

String line = **null**;

Vector<String> lines = **new** Vector<String>();

File file = **new** File(**this**.path);

BufferedReader br = **new** BufferedReader(**new** InputStreamReader(**new** FileInputStream(file)));

**while** ((line = br.readLine()) != **null**) {

lines.add(line);

}

br.close();

**return** lines;

}

}