МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Звіт з лабораторної роботи № 2

з предмету «Організація баз даних та знань»

Виконав:

Студент групи КН-36а

Кулик В.В.

Перевірили:

Козуля М.М.

Кизилов О.С.

Харків 2017

### Завдання на лабораторну роботу

### 1.1 Індивідуальне завдання

Спроектувати та реалізувати класи для представлення сутностей [попередньої лабораторної роботи](http://iwanoff.96.lt/oop_kn/LabTraining01.html). Слід створити два похідних класи від класу, який представляє основну сутність. Один клас повинен бути доповненим можливостями читання даних з відповідно підготовленого текстового файлу та запису цих даних в інший файл після сортування. Другий клас повинен реалізовувати читання даних з XML-документу, зберігати дані в структурах, які автоматично створюються за допомогою технології зв'язування даних, та запис даних в інший XML-документ після сортування.

Окрім роботи з файлами повинно бути реалізоване виведення результатів у консольне вікно.

### 1.2 Сортування цілих

Реалізувати програму читання з текстового файлу цілих додатних значень (числа розділені пробілами, слід читати до кінця файлу), занесення цих чисел у масив, сортування за зменшенням та за збільшенням суми цифр та зберігання обох результатів у двох нових текстових файлах. Перелічені дії реалізувати в окремій статичній функції. Для визначення порядку сортування створити класи, які реалізують інтерфейс Comparator.

### 1.3 Реалізація серіализації й десеріализації

Описати класи Студент і Академічна група (з полем - масивом студентів). Створити об'єкти, здійснити їх бінарну серіалізацію й десеріалізацію, а також серіалізацію й десеріалізацію в XML.

### 1.4 Список файлів усіх підкаталогів

Увести з клавіатури ім'я певної теки. Вивести на екран імена усіх файлів цієї теки, а також усіх файлів підкаталогів, їхніх підкаталогів тощо. Реалізувати виведення через рекурсивну функцію. Якщо тека не існує, вивести повідомлення про помилку.

**1.5 Робота з ZIP-архівом (додаткове завдання)**

Описати класи Студент і Академічна група (з полем - масивом студентів). Створити об'єкти, здійснити запис даних про студентів академічної групи в архів. В іншій програмі здійснити читання з архіву.

**Хід роботи**

### 1.1 Індивідуальне завдання

*Код програми 1.1:*

**Doc.java**

package seclab.f.ex;

public abstract class Doc {

private String name;

private String spec;

public Doc() {

}

public Doc(String name, String spec) {

this.name = name;

this.spec = spec;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getSpec() {

return spec;

}

public void setSpec(String spec) {

this.spec = spec;

}

abstract public Recept minOfVisitors();

abstract public int sumOfVisitors();

abstract public void addRecept(Recept recept);

abstract public void sortByVisitors();

abstract public void clearRecept();

abstract public void sortByDay();

abstract public Recept getRecept(int i);

abstract public int receptCount();

}

**DocArr.java**

package seclab.f.ex;

import java.util.Arrays;

public class DocArr extends Doc {

private Recept[] recepts;

public DocArr() {

// TODO Auto-generated constructor stub

}

public DocArr(String name,String spec, Recept[] recepts) {

super(name, spec);

this.recepts = recepts;

}

public void setRecepts(Recept[] recepts) {

this.recepts = recepts;

}

public Recept[] getRecepts() {

return recepts;

}

*@Override*

public Recept minOfVisitors() {

Recept min = recepts[0];

for (Recept h : recepts) {

if (h.getNumOfVisit() < min.getNumOfVisit())

min = h;

}

return min;

}

*@Override*

public int sumOfVisitors() {

int sum = 0;

for (Recept rec : recepts) {

sum += rec.getNumOfVisit();

}

return sum;

}

*@Override*

public void sortByVisitors() {

Arrays.*sort*(recepts, (a, b) -> Integer.*compare*(a.getNumOfVisit(), b.getNumOfVisit()));

}

*@Override*

public void sortByDay() {

Arrays.*sort*(recepts, (a, b) -> a.getDay().compareTo(b.getDay()));

}

*@Override*

public String toString() {

return "Doctor " + getName() + " " + getSpec() + "\n" + Arrays.*asList*(recepts);

}

*@Override*

public int receptCount() {

// TODO Auto-generated method stub

return 0;

}

*@Override*

public Recept getRecept(int i) {

return recepts[i];

}

*@Override*

public void addRecept(Recept recept) {

// TODO Auto-generated method stub

}

*@Override*

public void clearRecept() {

// TODO Auto-generated method stub

}

}

**DocArrTXT.java**

package seclab.f.ex;

import java.io.FileReader;

import java.io.FileWriter;

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class DocArrTXT extends DocArr {

public DocArrTXT() {

}

public DocArrTXT(String name, String spec, Recept[] recepts) {

super(name, spec, recepts);

}

public void toTxt(String name) {

try (FileWriter fReader = new FileWriter(name)) {

fReader.write(this.getName() + "\n");

fReader.write(this.getSpec() + "\n");

for (Recept rec : getRecepts()) {

fReader.write(rec.toString() + "\n");

}

} catch (Exception e) {

// TODO: handle exception

}

}

public DocArrTXT fromTxt() {

DocArrTXT sArr = new DocArrTXT();

try (Scanner scanner = new Scanner(new FileReader("Doctor.txt"))) {

sArr.setName(scanner.nextLine());

sArr.setSpec(scanner.nextLine());

List<Recept> sList = new ArrayList<>();

while (scanner.hasNext()) {

String[] string = scanner.nextLine().split(",");

Integer integer = Integer.parseInt(string[2].replaceAll(" ", ""));

sList.add(new Recept(integer, string[0], string[1]));

}

sArr.setRecepts(sList.toArray(new Recept[sList.size()]));

} catch (Exception e) {

e.printStackTrace();

}

return sArr;

}

}

**DocList.java**

package seclab.f.ex;

import java.util.Arrays;

import java.util.Collections;

import java.util.List;

public class DocList extends Doc{

private List<Recept> list;

public DocList() {

}

public DocList(String name, String spec, List<Recept> list) {

super(name, spec);

this.list = list;

}

public List<Recept> getList() {

return list;

}

public void setList(List<Recept> list) {

this.list = list;

}

@Override

public Recept minOfVisitors() {

Recept h = list.get(0);

for (Recept spec : list) {

if (h.getNumOfVisit() > spec.getNumOfVisit())

h = spec;

}

return h;

}

@Override

public int sumOfVisitors() {

int sum = 0;

for (Recept spec : list) {

sum += spec.getNumOfVisit();

}

return sum;

}

@Override

public void sortByVisitors() {

Collections.sort(list, (a, b) -> Integer.compare(a.getNumOfVisit(), b.getNumOfVisit()));

}

@Override

public void sortByDay() {

Collections.sort(list, (a, b) -> b.getDay().compareTo(a.getDay()));

}

@Override

public String toString() {

return "Doctor " + getName() + " " + getSpec() + "\n" + Arrays.asList(list.toArray());

}

@Override

public int receptCount() {

// TODO Auto-generated method stub

return 0;

}

@Override

public Recept getRecept(int i) {

// TODO Auto-generated method stub

return null;

}

@Override

public void addRecept(Recept recept) {

}

@Override

public void clearRecept() {

}

}

**DocListXML.java**

package seclab.f.ex;

import java.io.File;

import java.io.FileWriter;

import java.util.ArrayList;

import java.util.List;

import javax.xml.bind.JAXBContext;

import javax.xml.bind.Marshaller;

import javax.xml.bind.Unmarshaller;

public class DocListXML extends DocList{

private generated.Doc doc;

public DocListXML() {

}

public DocListXML(String name, String spec, List<Recept> list) {

super(name, spec, list);

}

public DocListXML fromXml(String filename) {

try {

JAXBContext jaxbContext=JAXBContext.newInstance(generated.Doc.class);

Unmarshaller unmarshaller=jaxbContext.createUnmarshaller();

doc =(generated.Doc) unmarshaller.unmarshal(new File("Doctor.xml"));

List<Recept> lis=new ArrayList<>();

for (generated.Doc.Recept recept : doc.getRecept()) {

lis.add(new Recept(recept.getNumOfVisit(),recept.getDay(),recept.getShift()));

}

this.setList(lis);

this.setName(doc.getName());

this.setSpec(doc.getSpec());

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

public void toXml(String name) {

try {

JAXBContext jaxbContext = JAXBContext.newInstance(generated.Doc.class);

Marshaller marshaller = jaxbContext.createMarshaller();

marshaller.setProperty(Marshaller.JAXB\_FORMATTED\_OUTPUT, Boolean.TRUE);

doc.setName(getName());

doc.setSpec(getSpec());

List<generated.Doc.Recept> list=new ArrayList<>();

for (Recept recept : getList()) {

list.add(new generated.Doc.Recept(recept.getNumOfVisit(),recept.getDay(),recept.getShift()));

}

marshaller.marshal(doc, new FileWriter(name));

} catch (Exception e) {

e.printStackTrace();

}

}

}

**Recept.java**

**package** seclab.f.ex;

**public** **class** **Recept** {

**private** **int** numOfVisit;

**private** **String** day;

**private** **String** shift;

**public** **Recept**() {}

**public** **Recept**(**int** numOfVisit, **String** day,**String** shift) {

**this**.numOfVisit = numOfVisit;

**this**.day = day;

**this**.shift = shift;

}

**public** **void** **setDay**(**String** day) {

**this**.day = day;

}

**public** **void** **setNumOfVisit**(**int** numOfVisit) {

**this**.numOfVisit = numOfVisit;

}

**public** **void** **setShift**(**String** shift) {

**this**.shift = shift;

}

**public** **String** **getDay**() {

**return** day;

}

**public** **int** **getNumOfVisit**() {

**return** numOfVisit;

}

**public** **String** **getShift**() {

**return** shift;

}

***@Override***

**public** **String** **toString**() {

**return** day + " , " + shift+","+numOfVisit;

}

}

**Test.java**

**package** seclab.f.ex;

**public** **class** **Test** {

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

**DocArrTXT** **sArr** = **new** DocArrTXT();

sArr = sArr.fromTxt();

sArr.sortByDay();

sArr.toTxt("sortbyDay.txt");

sArr.sortByVisitors();

sArr.toTxt("sortbyVisitors.txt");

**DocListXML** **list** = **new** DocListXML();

list.fromXml("Doctor.xml");

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

list.sortByDay();

list.toXml("sortbyDay.xml");

list.sortByVisitors();

list.toXml("sortbyVisitors.xml");

}

}

*Файл Doctor.txt містить такі дані:*

Сергей Курченко

Терапевт

15.05.16 , день , 17

16.05.16 , полдень , 26

02.05.16 , утро , 14

*Після запуску програми в файл sortbyDay.txt записуються такі дані:*

Сергей Курченко

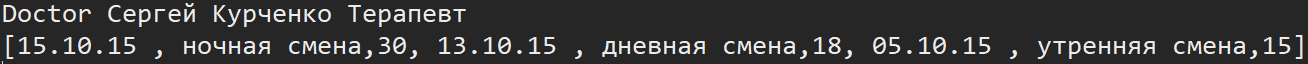
Терапевт

02.05.16 , утро ,14

15.05.16 , день ,17

16.05.16 , полдень ,26

*Виведення даних у консольне вікно:*



*Після запуску програми в файл sortbyDay.xml записуються такі дані:*

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<Doc name="Сергей Курченко" spec="Терапевт">

<Recept day="15.10.15" numOfVisit="30" shift="ночная смена"></Recept>

<Recept day="13.10.15" numOfVisit="18" shift="дневная смена"></Recept>

<Recept day="05.10.15" numOfVisit="15" shift="утренняя смена"></Recept>

</Doc>

*Файл Doctor.xml містить такі дані:*

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<Doc name="Сергей Курченко" spec="Терапевт">

<Recept day="15.10.15" numOfVisit="30" shift="ночная смена"></Recept>

<Recept day="13.10.15" numOfVisit="18" shift="дневная смена"></Recept>

<Recept day="05.10.15" numOfVisit="15" shift="утренняя смена"></Recept>

</Doc>

*Після запуску програми в файл sortbyVisitors.txt записуються такі дані:*

Сергей Курченко

Терапевт

02.05.16 , утро ,14

15.05.16 , день ,17

16.05.16 , полдень ,26

*Після запуску програми в файл sortbyVisitors.xml записуються такі дані:*

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<Doc name="Сергей Курченко" spec="Терапевт">

<Recept day="15.10.15" numOfVisit="30" shift="ночная смена"></Recept>

<Recept day="13.10.15" numOfVisit="18" shift="дневная смена"></Recept>

<Recept day="05.10.15" numOfVisit="15" shift="утренняя смена"></Recept>

</Doc>

### 1.2 Сортування цілих

*Код програми 1.2:*

**package seclab.sec\_ex;**

**import static java.lang.Math.abs;**

**import java.io.BufferedReader;**

**import java.io.FileReader;**

**import java.io.FileWriter;**

**import java.io.IOException;**

**import java.io.PrintWriter;**

**import java.util.ArrayList;**

**import java.util.Arrays;**

**import java.util.Comparator;**

**import java.util.Scanner;**

**public class Int {**

**public static Comparator<Integer> comareByAbsValues() {**

**class LocalComparator implements Comparator<Integer> {**

**@Override**

**public int compare(Integer d1, Integer d2) {**

**return -Integer.compare(abs(d1), abs(d2));**

**}**

**}**

**return new LocalComparator();**

**}**

**public static Comparator<Integer> comareByNorValues() {**

**class LocalComparator implements Comparator<Integer> {**

**@Override**

**public int compare(Integer d1, Integer d2) {**

**return Integer.compare(abs(d1), abs(d2));**

**}**

**}**

**return new LocalComparator();**

**}**

**void readWrite() {**

**try (BufferedReader reader = new BufferedReader(new FileReader("in.txt"))) {**

**String line;ArrayList<Integer> numbers = new ArrayList<>();Integer[] arr = {};**

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

**@SuppressWarnings("resource")**

**Scanner scanner = new Scanner(line);**

**while (scanner.hasNextInt()){**

**numbers.add(scanner.nextInt());**

**}**

**int i=numbers.size()-1;**

**while(i>=0){**

**if(numbers.get(i)<0){**

**numbers.remove(i);**

**}**

**i--;**

**}**

**}**

**arr = numbers.toArray(new Integer[numbers.size()]);**

**PrintWriter firstWriter = new PrintWriter(new FileWriter("out1.txt"));**

**PrintWriter secondWriter = new PrintWriter(new FileWriter("out2.txt"));**

**try {**

**Arrays.sort(arr,comareByNorValues());**

**for (int x : arr)**

**firstWriter.print(x + " ");**

**Arrays.sort(arr, comareByAbsValues());**

**for (int x : arr)**

**secondWriter.print(x + " ");**

**}**

**finally {**

**firstWriter.close();**

**secondWriter.close();}**

**}**

**catch (IOException ex) {**

**ex.printStackTrace();**

**}**

**}**

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

**new Int().readWrite();**

**}**

**}**

*Файл in.txt містить такі дані:*

5 24 13 22 1 8 11

*Після запуску програми в файл out1.txt записуються такі дані:*

1 5 8 11 13 22 24

*Після запуску програми в файл out2.txt записуються такі дані:*

24 22 13 11 8 5 1

### 1.3 Реалізація серіализації й десеріализації

*Код програми 1.3:*

**AcademGroup.java**

package lab2.seranddes;

import java.io.Serializable;

public class AcademGroup implements Serializable {

private static final long serialVersionUID = 8433147861334322335L;

private String name;

private Student[] students;

public AcademGroup(String name, Student... students) {

this.name = name;

this.students = students;

}

public AcademGroup() {

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Student[] getStudents() {

return students;

}

public void setStudents(Student... students) {

this.students = students;

}

}

**Student.java**

package lab2.seranddes;

import java.io.Serializable;

public class Student implements Serializable {

private static final long serialVersionUID = -6755942443306500892L;

private String name;

private int age;

public Student(String name, int age) {

this.name = name;

this.age = age;

}

public Student(){ }

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

}

**DataSerialization.java**

package lab2.seranddes;

import java.io.\*;

public class DataSerialization {

public static void main(String[] args) {

AcademGroup c = new AcademGroup("36A",

new Student("Gosha", 18),

new Student("Valera", 20),

new Student("Lesha", 19)

);

try (ObjectOutputStream out = new ObjectOutputStream(new FileOutputStream("AcademGroups.dat"))) {

out.writeObject(c);

}

catch (IOException e) {

e.printStackTrace();

};

}

}

**DataDeserialization.java**

package lab2.seranddes;

import java.io.\*;

public class DataDeserialization {

public static void main(String[] args) throws ClassNotFoundException {

try (ObjectInputStream in = new ObjectInputStream(new FileInputStream("AcademGroups.dat"))) {

AcademGroup academgroup = (AcademGroup) in.readObject();

System.out.println(academgroup.getName());

for (Student c : academgroup.getStudents()) {

System.out.println(c.getName() + " " + c.getAge());

}

}

catch (IOException e) {

e.printStackTrace();

};

}

}

**XMLSerialization.java**

package lab2.seranddes;

import java.beans.XMLEncoder;

import java.io.\*;

public class XMLSerialization {

public static void main(String[] args) {

AcademGroup c = new AcademGroup();

c.setName("36A");

Student a = new Student();

a.setName("Vlas Kurduk");

a.setAge(18);

Student b = new Student();

b.setName("Yulia Juya");

b.setAge(20);

c.setStudents(a, b);

try (XMLEncoder xmlEncoder = new XMLEncoder(new FileOutputStream("AcademGroup.xml"))) {

xmlEncoder.writeObject(a);

xmlEncoder.writeObject(b);

xmlEncoder.writeObject(c);

xmlEncoder.flush();

}

catch (IOException e) {

e.printStackTrace();

}

}

}

**XMLDeserialization.java**

package lab2.seranddes;

import java.beans.XMLDecoder;

import java.io.\*;

public class XMLDeserialization {

public static void main(String[] args) {

try (XMLDecoder xmlDecoder = new XMLDecoder(new FileInputStream("AcademGroup.xml"))) {

Student a = (Student) xmlDecoder.readObject();

Student b = (Student) xmlDecoder.readObject();

AcademGroup c = (AcademGroup) xmlDecoder.readObject();

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

for (Student j : c.getStudents()) {

System.out.println(j.getName() + " " + j.getAge());

}

}

catch (IOException e) {

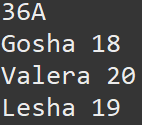
e.printStackTrace();

}

}

}

*Виведення даних у консольне вікно:*



*Після запуску програми в файл AcademGroup.xml записуються такі дані:*

<?xml version="1.0" encoding="UTF-8"?>

<java version="1.8.0\_141" class="java.beans.XMLDecoder">

<object class="lab2.seranddes.Student" id="Student0">

<void property="age">

<int>18</int>

</void>

<void property="name">

<string>Vlas Kurduk</string>

</void>

</object>

<object class="lab2.seranddes.Student" id="Student1">

<void property="age">

<int>20</int>

</void>

<void property="name">

<string>Yulia Juya</string>

</void>

</object>

<object class="lab2.seranddes.AcademGroup">

<void property="name">

<string>36A</string>

</void>

<void property="students">

<array class="lab2.seranddes.Student" length="2">

<void index="0">

<object idref="Student0"/>

</void>

<void index="1">

<object idref="Student1"/>

</void>

</array>

</void>

</object>

</java>

### 1.4  Список файлів усіх підкаталогів

*Код програми 1.4:*

**FileTest.java**

package seclab.four\_ex;

import java.io.File;

import java.io.IOException;

import java.util.Scanner;

public class FileTest {

public static void findDir(String path) throws IOException {

File file = new File(path);

//File[]list=file.listFiles();

for (File f : file.listFiles()) {

if (f.list() != null) {

System.out.println(f.getAbsolutePath());

findDir(f.getAbsolutePath());

}

else

System.out.println(f.getAbsolutePath());

try {

Thread.sleep(500);

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

public static void main(String[] args) throws IOException {

@SuppressWarnings("resource")

Scanner s=new Scanner(System.in);

System.out.println("Введите путь для поиска файлов: ");

String dirName=s.next();

File dir = new File(dirName);

if(!dir.isDirectory()){

do{

System.out.println("Данного пути не существует:");

dirName=s.next();

dir = new File(dirName);

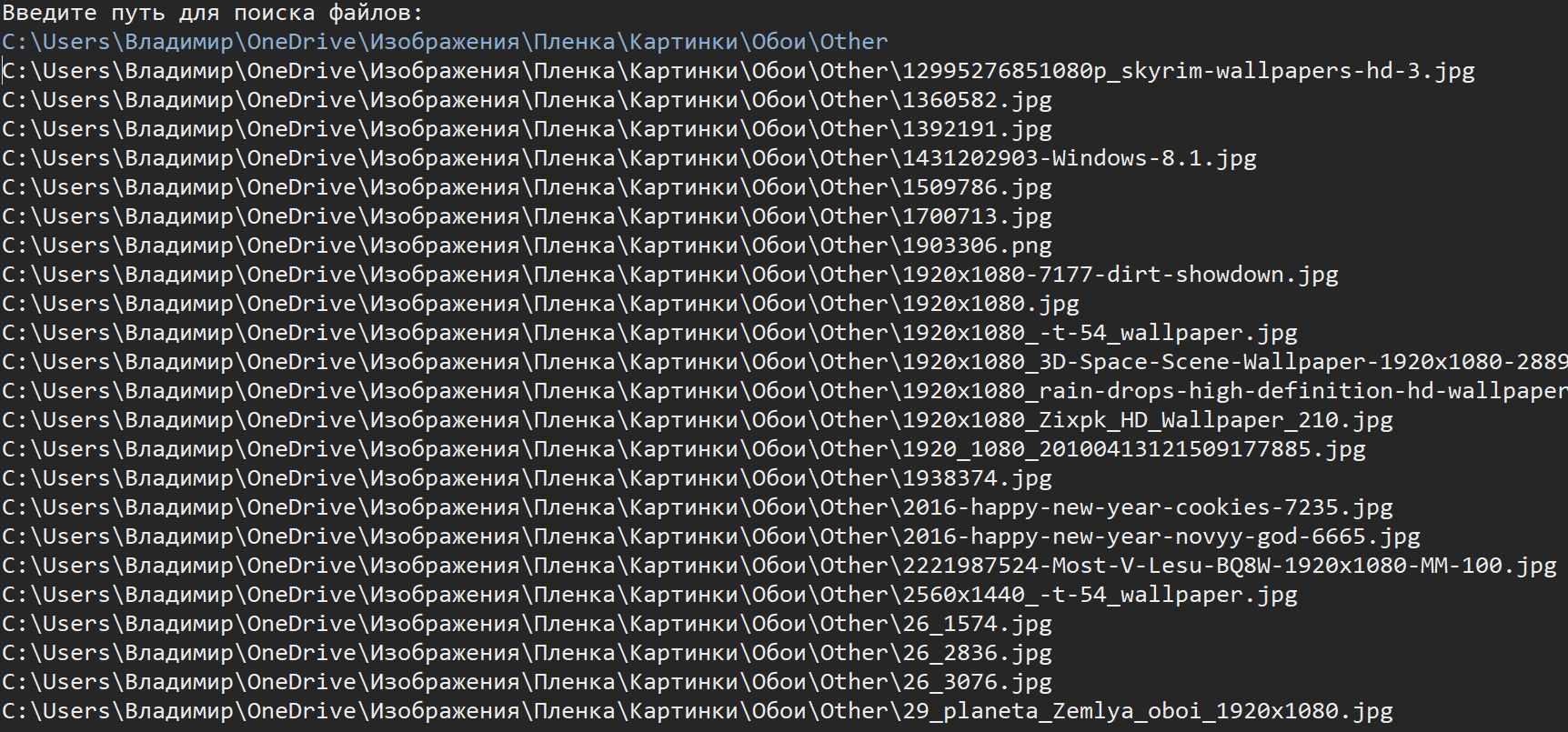
}while(!dir.isDirectory());}

findDir(dirName);

}

}

*Виведення даних у консольне вікно:*



**1.5 Робота з ZIP-архівом**

*Код програми 1.5:*

**Groupp.java**

**package seclab.fif\_ex;**

**import java.util.Arrays;**

**public class Groupp {**

**public StudentInf[] students;**

**public Groupp(StudentInf[] students) {**

**super();**

**this.students = students;**

**}**

**public StudentInf[] getStudents() {**

**return students;**

**}**

**public void setStudents(StudentInf[] students) {**

**this.students = students;**

**}**

**@Override**

**public String toString() {**

**return "Group members:\n" + Arrays.toString(students);**

**}**

**}**

**StudentInf.java**

package seclab.fif\_ex;

public class StudentInf {

public String name;

public String lastname;

public int yearsold;

public String getName() {

return name;

}

@Override

public String toString() {

return "Students name is " + name + "\nStudents lastname is " + lastname

+ "\nYearsold: " + yearsold;

}

public StudentInf(String name, String lastname, int yearsold) {

super();

this.name = name;

this.lastname = lastname;

this.yearsold = yearsold;

}

public void setName(String name) {

this.name = name;

}

public String getLastname() {

return lastname;

}

public void setLastname(String lastname) {

this.lastname = lastname;

}

public int getYearsold() {

return yearsold;

}

public void setYearsold(int yearsold) {

this.yearsold = yearsold;

}

}

**ZiP.java**

package seclab.fif\_ex;

import java.io.DataInputStream;

import java.io.DataOutputStream;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.FileWriter;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.zip.ZipEntry;

import java.util.zip.ZipInputStream;

import java.util.zip.ZipOutputStream;

public class ZiP {

public static void main(String[] args) {

StudentInf[] students = { new StudentInf("Vlada","Sushko",18), new StudentInf("Vlas","Antonenko",19), new StudentInf("Vadim","Gordychuk",18) };

Groupp groupp = new Groupp(students);

try (ZipOutputStream zOut = new ZipOutputStream(new FileOutputStream("36bGroup.zip"));

DataOutputStream out = new DataOutputStream(zOut)) {

for (StudentInf student : groupp.getStudents()) {

ZipEntry zipEntry = new ZipEntry(student.getLastname());

zOut.putNextEntry(zipEntry);

out.writeUTF(student.getName());

out.writeInt(student.getYearsold());

zOut.closeEntry();

}

FileWriter fw = new FileWriter("results.txt");

}

catch (IOException e) {

e.printStackTrace();

}

try (ZipInputStream zIn = new ZipInputStream(new FileInputStream("36bGroup.zip"));DataInputStream in = new DataInputStream(zIn)) {

ZipEntry entry;

while ((entry = zIn.getNextEntry()) != null) {

System.out.println("Last name: " + entry.getName());

System.out.println("Name: " + in.readUTF());

System.out.println("Years old: " + in.readInt());

System.out.println();

zIn.closeEntry();

}

}

catch (IOException e) {

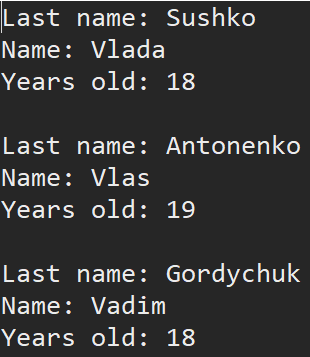
e.printStackTrace();

}

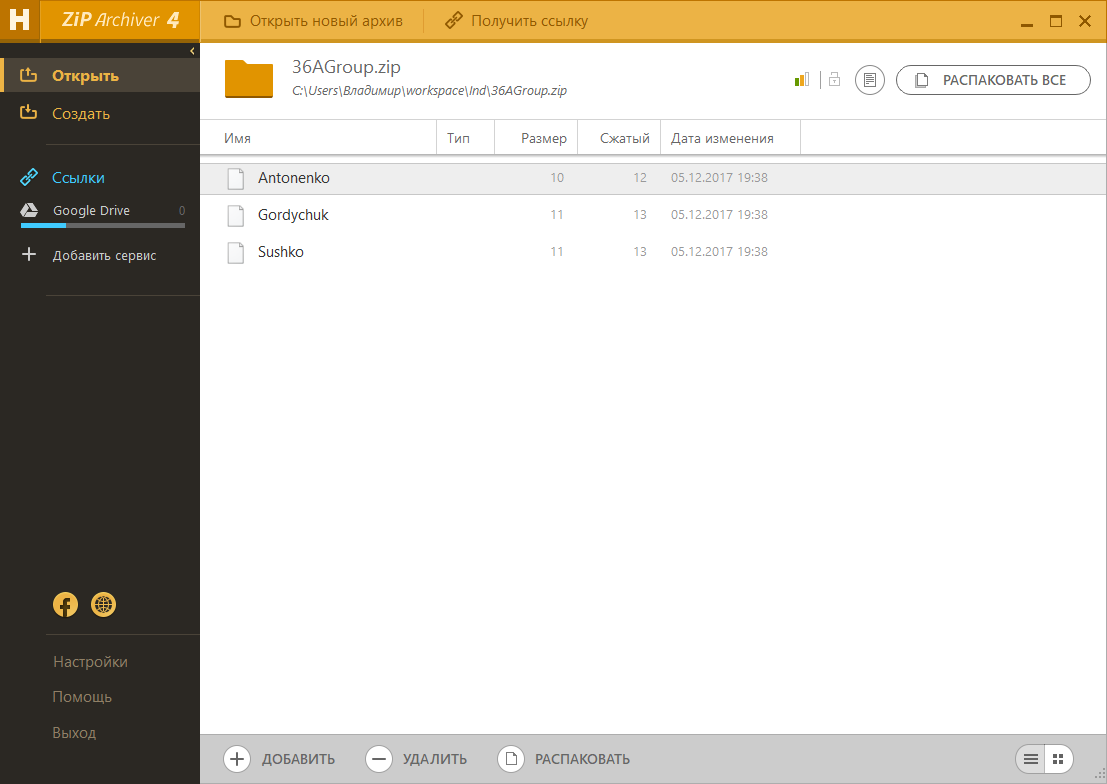
}

}

*Виведення даних у консольне вікно:*



*Після запуску програми створюється архів 36AGroup.zip:*



**Висновок:**

У даній роботі ми за допомогою мови програмування Java та використовуючи попередній досвід розробили різного роду програми, під час розробки яких ми навчилися працювати з винятками та файлами в Java.