**JAVA LAB – 9**

**FILE IO , Collections and Serialization**

**19BAI1157**

**Yash Tripathi**

**Question>**

Write a java IO and collection program to

* Identify the numbers of births in particular region from the given input file named "births-deaths-by-region".
* identify the number of death during the period 2020.
* Identify the total umber of regions
* Identify the year which has more birth counts

Save the object of your class as "lab9.dat" and perform deserialization

**Solution>**

Dataread.java

import java.io.\*;

import java.util.ArrayList;

import java.util.List;

import java.util.\*;

public class DataRead {

public static void main(String[] args) {

String data = ("/Users/yashtripathi/Documents/OneDrive/yash docs/Work/Collage/Semister 5/Java Lab/LAB\_10/data.csv");

try{

List<regionalStats> df= readCsvStats(data);

staticMethods Solutions = new staticMethods();

Solutions.Q1(df);

Solutions.Q2(df);

Solutions.Q3(df);

Solutions.Q4(df);

System.out.println("Saving the ARRAY of custom objects using seriisation reading it again to solve question ");

serelise.save(df);

List<regionalStats>obj = deSerelise.read();

Solutions.Q4(obj);

}

catch(IOException e)

{

System.out.println(e+"1");

}

catch(ClassNotFoundException e)

{

System.out.println(e);

}

}

static List<regionalStats> readCsvStats(String name) throws FileNotFoundException {

List<regionalStats> dfList = new ArrayList<>();

try {

File data = new File(name);

Scanner df = new Scanner(data);

String line = df.nextLine();

line = df.nextLine();

while (line != null) {

String[] attributes = line.split(",");

regionalStats record = createRecord(attributes);

dfList.add(record);

line = df.nextLine();

}

df.close();

} catch (Exception e) {

System.out.println(e);

}

return dfList;

}

static regionalStats createRecord(String[] metadata)

{

int period = Integer.parseInt(metadata[0]);

Boolean birth;

if(new String(metadata[1]).equals(new String("Births")) ) { birth = true;}

else{birth = false;}

String region = metadata[2];

int count = Integer.parseInt(metadata[3]);

return new regionalStats(period, birth, region, count);

}

}

**RegionalStats.java**

import java.io.Serializable;

public class regionalStats implements Serializable {

int period;

Boolean birth;

String region;

int count;

regionalStats(

int period,

Boolean birth,

String region,

int count

)

{

this.period = period;

this.birth = birth;

this.region = region;

this.count = count;

}

@Override public String toString() { return "Record [period=" + period + ", birth=" + birth + ", region=" + region + ", count="+count+"]"; }

}

**StaticMeathods.java**

import java.util.ArrayList;

import java.util.HashSet;

import java.util.List;

import java.util.Scanner;

public class staticMethods {

void Q1(List<regionalStats> df) {

System.out.println("---------QUESTION\_1------------");

System.out.println(

"Q.Identify the numbers of births in particular region from the given input file named \"births-deaths-by-region\".");

int answer = 0;

Scanner input = new Scanner(System.in);

System.out.println("Enter Region name:- ");

String Region = input.nextLine();

for (regionalStats d : df) {

if (d.region.equals(Region)) {

if (d.birth) {

answer = answer + d.count;

}

}

}

if (answer == 0) {

System.out.println("Region Not Found");

} else {

System.out.println("Answer:- " + answer);

}

System.out.println("---------END QUESTION\_1------------");

input.close();

}

void Q2(List<regionalStats> df) {

System.out.println("---------QUESTION\_2------------");

System.out.println("Q.identify the number of death during the period 2020.");

int answer = 0;

for (regionalStats d : df) {

if (d.period == 2020) {

if (!d.birth) {

answer = answer + d.count;

}

}

}

if (answer == 0) {

System.out.println("Region Not Found");

} else {

System.out.println("Answer:- " + answer);

}

System.out.println("---------END QUESTION\_2------------");

}

void Q3(List<regionalStats> df) {

System.out.println("---------QUESTION\_3------------");

System.out.println("Q.Identify the total number of regions.");

int answer = 0;

ArrayList<String> ArrList = new ArrayList<String>();

for (regionalStats d : df) {

ArrList.add(d.region);

}

HashSet<String> hset = new HashSet<String>(ArrList);

answer = hset.size();

if (answer == 0) {

System.out.println("Region Not Found");

} else {

System.out.println("Answer:- " + answer);

}

System.out.println("---------END QUESTION\_3------------");

}

void Q4(List<regionalStats> df) {

System.out.println("---------QUESTION\_4------------");

System.out.println("Q.Identify the year which has more birth counts.");

int answer = 0;

int BestTotal = 0;

int currentTotal = 0;

int currentYear = 2000;

for (regionalStats d : df) {

if(currentYear == d.period)

{

if (d.birth) {

currentTotal = currentTotal + d.count;

}

}

else{

if(currentTotal > BestTotal)

{

answer = currentYear;

BestTotal = currentTotal;

}

currentTotal = 0;

currentYear = d.period;

if (d.birth) {

currentTotal = currentTotal + d.count;

}

}

}

System.out.println("Answer:- " + answer);

System.out.println("---------END QUESTION\_4------------");

}

}

**Serilise.java**

import java.io.FileNotFoundException;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.ObjectOutputStream;

import java.util.List;

public class serelise {

public static void save(List<regionalStats> obj) throws FileNotFoundException, IOException {

ObjectOutputStream out = new ObjectOutputStream(new FileOutputStream("lab9.ser"));

// regionalStats new1 = new regionalStats(2005, true, "Calcutta", 2314);

out.writeObject(obj);

out.close();

}

}

**Deserilise.java**

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.io.ObjectInputStream;

import java.util.List;

public class deSerelise {

public static List<regionalStats> read() throws FileNotFoundException, IOException, ClassNotFoundException {

ObjectInputStream in = new ObjectInputStream(new FileInputStream("lab9.ser"));

System.out.println();

List<regionalStats> data = (List<regionalStats>)in.readObject();

in.close();

return data;

}

}

**Screenshots>**

**Text

Description automatically generated**