Countries Of The World App 2.0 - Java

CS 3310 – Dr. Donna Kaminski

Index

[TheLog.txt 2](#_Toc382465373)

[CountryIndex.txt 6](#_Toc382465374)

[Main 7](#_Toc382465375)

[Setup 8](#_Toc382465376)

[UserApp 9](#_Toc382465377)

[RawData 11](#_Toc382465378)

[TransData 13](#_Toc382465379)

[TheLog 16](#_Toc382465380)

[CountryData 19](#_Toc382465381)

[CountryIndex 23](#_Toc382465382)

[CountryDataRec 27](#_Toc382465383)

[Node 30](#_Toc382465384)

# FILE STATUS > TheLog FILE opened.

CODE STATUS > Setup started.

FILE STATUS > RawData FILE opened.

FILE STATUS > CountryData FILE opened.

FILE STATUS > RawData FILE closed.

CODE STATUS > Snapshot CountryData started.

N: 26

[RRN] ID CDE NAME------------ CONTINENT---- ------AREA ---POPULATION LIFE

[001] 030 EGY Egypt Africa 1,001,449 68,470,000 63.3

[002] 032 GBR United Kingdom Europe 242,900 59,623,400 77.7

[003] 060 LBN Lebanon Asia 10,400 3,282,000 71.3

[004] 001 XVI Hex Land North America 1 1,966,514,816 0.0

[005] 035 QAT Qatar Asia 11,000 599,000 72.4

[006] 006 USA United States North America 9,363,520 278,357,000 77.1

[007] 037 WSM Samoa Oceania 2,831 180,000 69.2

[008] 120 RUS Russian Federati Europe 17,075,400 146,934,000 67.2

[009] 039 HTI Haiti North America 27,750 8,222,000 49.2

[010] 010 NIC Nicaragua North America 130,000 5,074,000 68.7

[011] 069 SOM Somalia Africa 637,657 10,097,000 46.2

[012] 012 MEX Mexico North America 1,958,201 98,881,000 71.5

[013] 013 JPN Japan Asia 377,829 126,714,000 80.7

[014] 043 VEN Venezuela South America 912,050 24,170,000 73.1

[015] 072 IND India Asia 3,287,263 1,013,662,000 62.5

[016] 042 OMN Oman Asia 309,500 2,542,000 71.8

[017] 102 KEN Kenya Africa 580,367 30,080,000 48.0

[018] 132 ZWE Zimbabwe Africa 390,757 11,669,000 37.8

[019] 019 FRA France Europe 551,500 59,225,700 78.8

[020] 020 POL Poland Europe 323,250 38,653,600 73.2

[021] 059 ATA Antarctica Antarctica 13,120,000 0 0.0

[022] EMPTY

[023] 023 CHN China Asia 9,572,900 1,277,558,000 71.4

[024] 053 DEU Germany Europe 357,022 82,164,700 77.4

[025] EMPTY

[026] EMPTY

[027] 027 BEL Belgium Europe 30,518 10,239,000 77.8

[028] EMPTY

[029] 029 TCA Turks and Caicos North America 430 17,000 73.3

[030] 090 YEM Yemen Asia 527,968 18,112,000 59.8

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

CODE STATUS > Snapshot CountryData finished - 26 countries displayed.

FILE STATUS > CountryData FILE closed.

CODE STATUS > Snapshot CountryIndex started.

CODE INDEX > MAX\_N\_HOME\_LOC: 20, nHome: 11, nColl: 15

[SUB] CODE | DRP | LINK |

[000] JPN | 013 | 034 |

[001] EMPTY

[002] EMPTY

[003] EMPTY

[004] MEX | 012 | 029 |

[005] EMPTY

[006] EMPTY

[007] EMPTY

[008] LBN | 003 | -01 |

[009] SOM | 011 | -01 |

[010] KEN | 017 | 032 |

[011] EGY | 001 | -01 |

[012] CHN | 023 | 026 |

[013] EMPTY

[014] OMN | 016 | -01 |

[015] USA | 006 | -01 |

[016] EMPTY

[017] YEM | 030 | 028 |

[018] NIC | 010 | -01 |

[019] EMPTY

[020] FRA | 019 | -01 |

[021] BEL | 027 | -01 |

[022] GBR | 002 | -01 |

[023] HTI | 009 | 021 |

[024] VEN | 014 | 022 |

[025] POL | 020 | 020 |

[026] IND | 015 | 024 |

[027] DEU | 024 | 025 |

[028] WSM | 007 | -01 |

[029] XVI | 004 | 023 |

[030] QAT | 005 | 027 |

[031] ZWE | 018 | -01 |

[032] RUS | 008 | 031 |

[033] TCA | 029 | 030 |

[034] ATA | 021 | 033 |

CODE STATUS > Snapshot CountryIndex finished - 26 countries displayed.

FILE STATUS > IndexBackup File opened.

FILE STATUS > IndexBackup File backed up and closed.

CODE STATUS > Setup finished - 26 countries processed.

FILE STATUS > TheLog FILE closed.

FILE STATUS > TheLog FILE opened.

CODE STATUS > UserApp started.

FILE STATUS > TransData FILE opened.

FILE STATUS > CountryData FILE opened.

FILE STATUS > IndexBackup File opened.

FILE STATUS > IndexBackup File restored and closed.

SC MEX

12 MEX Mexico North America 1,958,201 98,881,000 71.5

1 data records read

0 index nodes visited

SC JPN

13 JPN Japan Asia 377,829 126,714,000 80.7

1 data records read

0 index nodes visited

SC YEM

90 YEM Yemen Asia 527,968 18,112,000 59.8

1 data records read

0 index nodes visited

SC BEL

27 BEL Belgium Europe 30,518 10,239,000 77.8

1 data records read

3 index nodes visited

SI 12

12 MEX Mexico North America 1,958,201 98,881,000 71.5

1 data records read

SI 59

59 ATA Antarctica Antarctica 13,120,000 0 0.0

23 data records read

SI 30

30 EGY Egypt Africa 1,001,449 68,470,000 63.3

2 data records read

SI 90

90 YEM Yemen Asia 527,968 18,112,000 59.8

1 data records read

SI 1

1 XVI Hex Land North America 1 1,966,514,816 0.0

4 data records read

SI 29

29 TCA Turks and Caicos North America 430 17,000 73.3

1 data records read

SI 102

102 KEN Kenya Africa 580,367 30,080,000 48.0

6 data records read

SC ATA

59 ATA Antarctica Antarctica 13,120,000 0 0.0

1 data records read

1 index nodes visited

SC LBN

60 LBN Lebanon Asia 10,400 3,282,000 71.3

1 data records read

0 index nodes visited

SC WSM

37 WSM Samoa Oceania 2,831 180,000 69.2

1 data records read

1 index nodes visited

SC NPJ

SORRY, no country with that id.

6 index nodes visited

SC NLB

SORRY, no country with that id.

0 index nodes visited

SC EMY

SORRY, no country with that id.

1 index nodes visited

SC QQQ

SORRY, no country with that id.

0 index nodes visited

SC IQQ

SORRY, no country with that id.

0 index nodes visited

DI 60

SORRY, not yet working

DI 22

SORRY, not yet working

DC USA

SORRY, not yet working.

DC FRA

SORRY, not yet working.

DC WMU

SORRY, not yet working.

SI 23

23 CHN China Asia 9,572,900 1,277,558,000 71.4

1 data records read

SI 53

53 DEU Germany Europe 357,022 82,164,700 77.4

2 data records read

SI 83

SORRY, no country with that id.

SI 89

SORRY, no country with that id.

SI 52

SORRY, no country with that id.

SI 0

SORRY, no country with that id.

SI 999

SORRY, no country with that id.

SI 131

SORRY, no country with that id.

SI 58

SORRY, no country with that id.

SI 32

32 GBR United Kingdom Europe 242,900 59,623,400 77.7

1 data records read

SI 74

SORRY, no country with that id.

SI 927

SORRY, no country with that id.

SI 925

SORRY, no country with that id.

SC JOR

SORRY, no country with that id.

3 index nodes visited

SC GRC

SORRY, no country with that id.

0 index nodes visited

SC MYS

SORRY, no country with that id.

0 index nodes visited

FILE STATUS > TransData FILE closed.

CODE STATUS > Snapshot CountryData started.

N: 26

[RRN] ID CDE NAME------------ CONTINENT---- ------AREA ---POPULATION LIFE

[001] 030 EGY Egypt Africa 1,001,449 68,470,000 63.3

[002] 032 GBR United Kingdom Europe 242,900 59,623,400 77.7

[003] 060 LBN Lebanon Asia 10,400 3,282,000 71.3

[004] 001 XVI Hex Land North America 1 1,966,514,816 0.0

[005] 035 QAT Qatar Asia 11,000 599,000 72.4

[006] 006 USA United States North America 9,363,520 278,357,000 77.1

[007] 037 WSM Samoa Oceania 2,831 180,000 69.2

[008] 120 RUS Russian Federati Europe 17,075,400 146,934,000 67.2

[009] 039 HTI Haiti North America 27,750 8,222,000 49.2

[010] 010 NIC Nicaragua North America 130,000 5,074,000 68.7

[011] 069 SOM Somalia Africa 637,657 10,097,000 46.2

[012] 012 MEX Mexico North America 1,958,201 98,881,000 71.5

[013] 013 JPN Japan Asia 377,829 126,714,000 80.7

[014] 043 VEN Venezuela South America 912,050 24,170,000 73.1

[015] 072 IND India Asia 3,287,263 1,013,662,000 62.5

[016] 042 OMN Oman Asia 309,500 2,542,000 71.8

[017] 102 KEN Kenya Africa 580,367 30,080,000 48.0

[018] 132 ZWE Zimbabwe Africa 390,757 11,669,000 37.8

[019] 019 FRA France Europe 551,500 59,225,700 78.8

[020] 020 POL Poland Europe 323,250 38,653,600 73.2

[021] 059 ATA Antarctica Antarctica 13,120,000 0 0.0

[022] EMPTY

[023] 023 CHN China Asia 9,572,900 1,277,558,000 71.4

[024] 053 DEU Germany Europe 357,022 82,164,700 77.4

[025] EMPTY

[026] EMPTY

[027] 027 BEL Belgium Europe 30,518 10,239,000 77.8

[028] EMPTY

[029] 029 TCA Turks and Caicos North America 430 17,000 73.3

[030] 090 YEM Yemen Asia 527,968 18,112,000 59.8

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

CODE STATUS > Snapshot CountryData finished - 26 countries displayed.

FILE STATUS > CountryData FILE closed.

CODE STATUS > Snapshot CountryIndex started.

CODE INDEX > MAX\_N\_HOME\_LOC: 20, nHome: 0, nColl: 0

[SUB] CODE | DRP | LINK |

[000] JPN | 013 | 034 |

[001] EMPTY

[002] EMPTY

[003] EMPTY

[004] MEX | 012 | 029 |

[005] EMPTY

[006] EMPTY

[007] EMPTY

[008] LBN | 003 | -01 |

[009] SOM | 011 | -01 |

[010] KEN | 017 | 032 |

[011] EGY | 001 | -01 |

[012] CHN | 023 | 026 |

[013] EMPTY

[014] OMN | 016 | -01 |

[015] USA | 006 | -01 |

[016] EMPTY

[017] YEM | 030 | 028 |

[018] NIC | 010 | -01 |

[019] EMPTY

[020] FRA | 019 | -01 |

[021] BEL | 027 | -01 |

[022] GBR | 002 | -01 |

[023] HTI | 009 | 021 |

[024] VEN | 014 | 022 |

[025] POL | 020 | 020 |

[026] IND | 015 | 024 |

[027] DEU | 024 | 025 |

[028] WSM | 007 | -01 |

[029] XVI | 004 | 023 |

[030] QAT | 005 | 027 |

[031] ZWE | 018 | -01 |

[032] RUS | 008 | 031 |

[033] TCA | 029 | 030 |

[034] ATA | 021 | 033 |

CODE STATUS > Snapshot CountryIndex finished - 26 countries displayed.

FILE STATUS > IndexBackup File opened.

FILE STATUS > IndexBackup File backed up and closed.

CODE STATUS > UserApp finished - 40 transactions processed.

FILE STATUS > TheLog FILE closed.

# JPN,13,34

,,-1

,,-1

,,-1

MEX,12,29

,,-1

,,-1

,,-1

LBN,3,-1

SOM,11,-1

KEN,17,32

EGY,1,-1

CHN,23,26

,,-1

OMN,16,-1

USA,6,-1

,,-1

YEM,30,28

NIC,10,-1

,,-1

FRA,19,-1

BEL,27,-1

GBR,2,-1

HTI,9,21

VEN,14,22

POL,20,20

IND,15,24

DEU,24,25

WSM,7,-1

XVI,4,23

QAT,5,27

ZWE,18,-1

RUS,8,31

TCA,29,30

ATA,21,33

**package** edu.wmich.cs3310.asgn1;

**import** java.io.File;

**import** java.io.IOException;

/\*\*

\* Countries Of The World App 1.0

\* Creates table for managing contries information.

\* **@author** Caleb Viola

\*/

# **public** **class** Main {

/\*\*

\* Deletes binary file if exists,

\* calls Setup and Userapp according to

\* A2DemoSpecs.

\* **@param** args

\* **@throws** IOException

\*/

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

File file = **new** File("CountryData.bin");

**if** (file.exists())

file.delete();

Setup.*Table*("A3");

**for** (**int** i = 8; i <= 8; i++)

UserApp.*Table*(i);

}

**package** edu.wmich.cs3310.asgn1;

**import** java.io.IOException;

/\*\*

\* Class containing method to setup CDT from the RawData files.

\* **@author** Caleb Viola

\*/

# **public** **class** Setup {

/\*\*

\* Method intented to setup CDT from the RawData files.

\* **@throws** IOException

\*/

**public** **static** **void** Table(String fileNameSufix) **throws** IOException {

TheLog tL = **new** TheLog(**false**);

tL.displayThis("CODE STATUS > Setup started.");

RawData rD = **new** RawData(fileNameSufix, tL);

CountryData cD = **new** CountryData(tL, "CountryData.bin");

CountryIndex cI= **new** CountryIndex(tL);

**int** rrn, temp = 0;

rD.grabCountry();

**while** (!rD.doneWithInput) {

rrn = cD.insertCountry(rD.getID(), rD.getCode(), rD.getName(),

rD.getContinent(), rD.getArea(), rD.getPopulation(),

rD.getLifeExpectancy(), tL, **false**);

cI.insertCodeInIndex(rD.getCode(), rrn, tL, **false**);

temp++;

rD.grabCountry();

}

rD.finishUp(tL);

cD.finishUp(tL, **true**);

cI.finishUp(tL, **true**);

tL.displayThis("CODE STATUS > Setup finished - " + temp

+ " countries processed.");

tL.finishUp();

}

}

**package** edu.wmich.cs3310.asgn1;

**import** java.io.IOException;

/\*\*

\* Class containing method to manage table modifinations from the TransData

\* files.

\* **@author** Caleb Viola

\*/

# **public** **class** UserApp {

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Method intented to manage table modifinations from the TransData files.

\* **@param** fileNameSufix

\* **@throws** IOException

\*/

**public** **static** **void** Table(**int** fileNameSufix)**throws** IOException {

TheLog tL = **new** TheLog(**true**);

tL.displayThis("CODE STATUS > UserApp started.");

TransData tD = **new** TransData(fileNameSufix, tL);

CountryData cD = **new** CountryData(tL, "CountryData.bin");

CountryIndex cI= **new** CountryIndex(tL);

cI.restoreIndex(tL);

**int** rrn, temp = 0;

**while** (!tD.doneWithFile) {

tD.grabCommand(tL);

**if** (!tD.doneWithFile) {

**switch** (tD.transCode()) {

**case** "SI":

**if** (tD.getID() == -1) {

tL.displayThis("SI ERROR");

tL.displayThis(tL.sorry3());

temp++;

**break**;

}

tL.displayThis("SI " + tD.getID());

cD.selectByID(tD.getID(), tL, **false**);

temp++;

**break**;

**case** "SC":

tL.displayThis("SC " + tD.getCode());

cD.selectByID(cI.selectByCode(tD.getCode(), tL), tL, **true**);

tL.displayThis(String.*format*(" %s index nodes visited",

cI.getNodRead()));

temp++;

**break**;

**case** "DI":

**if** (tD.getID() == -1) {

tL.displayThis("DI ERROR");

tL.displayThis(tL.sorry3());

temp++;

**break**;

}

tL.displayThis("DI " + tD.getID());

cD.deleteByID(tD.getID(), tL);

temp++;

**break**;

**case** "DC":

**if** (tD.getID() == -1) {

tL.displayThis("DC ERROR");

tL.displayThis(tL.sorry3());

temp++;

**break**;

}

tL.displayThis("DC " + tD.getCode());

cI.deleteByCode(tD.getCode(), tL);

temp++;

**break**;

**case** "IN":

tL.displayThis("IN " + tD.getID() + "," + tD.getCode()

+ "," + tD.getName() + "," + tD.getArea() + ","

+ tD.getPopulation() + "," +

tD.getLifeExpectancy());

rrn = cD.insertCountry(tD.getID(), tD.getCode(),

tD.getName(), tD.getContinent(), tD.getArea(),

tD.getPopulation(), tD.getLifeExpectancy(), tL,

**true**);

cI.insertCodeInIndex(tD.getCode(), rrn, tL, **true**);

temp++;

**break**;

**default**:

tL.displayThis(tD.transCode() + "\n"

+ " ERROR, invalid command.");

temp++;

}

}

}

tD.finishUp(tL);

cD.finishUp(tL, **true**);

cI.finishUp(tL, **true**);

tL.displayThis("CODE STATUS > UserApp finished - " + temp

+ " transactions processed.");

tL.finishUp();

}

}

**package** edu.wmich.cs3310.asgn1;

**import** java.io.File;

**import** java.io.IOException;

**import** java.util.Scanner;

/\*\*

\* Intended to obtain information from the RawData files for the Setup class.

\* Countries Of The World App 1.0

\* **@author** Caleb Viola

\*/

# **public** **class** RawData {

**private** Scanner input;

**private** **short** id;

**private** String code;

**private** String name;

**private** String continent;

**private** **int** area;

**private** **long** population;

**private** **float** lifeExpectancy;

**boolean** doneWithInput = **false**;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Contructor to open RawData and create log entry.

\* **@param** fileNameSufix Name of RawData file to open.

\* **@param** tl TheLog object

\* **@throws** IOException

\*/

**public** RawData(String fileNameSufix, TheLog tl) **throws** IOException {

File file = **new** File("RawData" + fileNameSufix + ".csv");

tl.displayThis("FILE STATUS > RawData FILE opened.");

input = **new** Scanner(file);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Obtain information from one line in the RawData file.

\*/

**public** **void** grabCountry() {

**if** (input.hasNextLine()) {

String temp = input.nextLine();

temp = temp.substring(30, temp.length() - 2).replace("'", "");

String[] fields = temp.split(",");

id = Short.*parseShort*(fields[0]);

code = fields[1];

name = fields[2];

continent = fields[3];

area = Integer.*parseInt*(fields[5]);

population = Integer.*parseInt*(fields[7]);

lifeExpectancy = Float.*parseFloat*(fields[8]);

} **else**

doneWithInput = **true**;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for id.

\* **@return**

\*/

**public** **short** getID(){

**return** id;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for code.

\* **@return**

\*/

**public** String getCode(){

**return** code;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for name.

\* **@return**

\*/

**public** String getName(){

**return** name;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for continent.

\* **@return**

\*/

**public** String getContinent(){

**return** continent;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for area.

\* **@return**

\*/

**public** **int** getArea(){

**return** area;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for population.

\* **@return**

\*/

**public** **long** getPopulation(){

**return** population;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for lifeExpectancy.

\* **@return**

\*/

**public** **float** getLifeExpectancy(){

**return** lifeExpectancy;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Close file and create log entry.

\* **@param** tl TheLog object

\* **@throws** IOException

\*/

**public** **void** finishUp(TheLog tl) **throws** IOException{

input.close();

tl.displayThis("FILE STATUS > RawData FILE closed.");

}

}

**package** edu.wmich.cs3310.asgn1;

**import** java.io.File;

**import** java.io.IOException;

**import** java.util.Scanner;

/\*\*

\* Intended to obtain information for UserApp in Main from the TransData files

\* in order to create table.

\* **@author** Caleb Viola

\*/

# **public** **class** TransData {

**private** Scanner input;

**private** **short** id;

**private** String code;

**private** String name;

**private** String continent;

**private** String command;

**private** **int** area;

**private** **long** population;

**private** **float** lifeExpectancy;

**boolean** doneWithFile = **false**;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Constructor to open a TransData file and create log entry.

\* **@param** fileNameSufix number of TransData file to read.

\* **@param** tl TheLog object

\* **@throws** IOException

\*/

**public** TransData(**int** fileNameSufix, TheLog tl) **throws** IOException {

File file = **new** File("TransData" + fileNameSufix + ".txt");

tl.displayThis("FILE STATUS > TransData FILE opened.");

input = **new** Scanner(file);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Obtain command from a line in the TransData file.

\* **@param** tl TheLog object

\* **@throws** IOException

\*/

**public** **void** grabCommand(TheLog tl) **throws** IOException {

**if** (input.hasNextLine()) {

String temp = input.nextLine();

command = temp.substring(0, 2);

**if** (command.equals("IN")) {

temp = temp.substring(33, temp.length() - 2).replace("'",

"");

String[] fields = temp.split(",");

id = Short.*parseShort*(fields[0]);

code = fields[1];

name = fields[2];

continent = fields[3];

area = Integer.*parseInt*(fields[5]);

population = Integer.*parseInt*(fields[7]);

lifeExpectancy = Float.*parseFloat*(fields[8]);

}

**else** **if** (command.equals("SI") || command.equals("DI")) {

**if** (temp.length() > 2) {

**try** {

id = Short.*parseShort*(temp.substring(3,

temp.length()).trim());

} **catch** (NumberFormatException e) {

id = -1;

}

} **else**

id = -1;

}**else** **if** (command.equals("SC") || command.equals("DC")) {

**if** (temp.length() > 2)

code = temp.substring(3, temp.length()).trim();

}

} **else**

doneWithFile = **true**;

}

/\*\*

\* Getter for acquired command.

\* **@return**

\*/

**public** String transCode() {

**return** command;

}

/\*\*

\* Getter for id.

\* **@return**

\*/

**public** **short** getID() {

**return** id;

}

/\*\*

\* Getter for code.

\* **@return**

\*/

**public** String getCode() {

**return** code;

}

/\*\*

\* Getter for name.

\* **@return**

\*/

**public** String getName() {

**return** name;

}

/\*\*

\* Getter for continent.

\* **@return**

\*/

**public** String getContinent() {

**return** continent;

}

/\*\*

\* Getter for area.

\* **@return**

\*/

**public** **int** getArea() {

**return** area;

}

/\*\*

\* Getter for population.

\* **@return**

\*/

**public** **long** getPopulation() {

**return** population;

}

/\*\*

\* Getter for lifeExpectancy.

\* **@return**

\*/

**public** **float** getLifeExpectancy() {

**return** lifeExpectancy;

}

/\*\*

\* Close file and creat log entry.

\* **@param** tl TheLog object

\* **@throws** IOException

\*/

**public** **void** finishUp(TheLog tl) **throws** IOException {

input.close();

tl.displayThis("FILE STATUS > TransData FILE closed.");

}

}

**package** edu.wmich.cs3310.asgn1;

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

**import** java.text.DecimalFormat;

/\*\*

\* Writes log entries to a .txt file

\* Countries Of The World App 1.0

\* **@author** Caleb Viola

\*/

# **public** **class** TheLog {

**private** PrintWriter tL;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Intended to set up theLog according to boolean for append.

\* **@param** append Decides append mode for PrintWriter.

\* **@throws** IOException

\*/

**public** TheLog(**boolean** append) **throws** IOException {

**if** (!append)

tL = **new** PrintWriter("TheLog.txt");

**else**

tL = **new** PrintWriter(**new** FileOutputStream(

**new** File("TheLog.txt"), append));

tL.println("FILE STATUS > TheLog FILE opened.");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Preformatted string with Snapshot format for printing country

data.

\* **@param** index

\* **@param** code

\* **@param** name

\* **@param** continent

\* **@param** area

\* **@param** population

\* **@param** life

\* **@param** LCh

\* **@param** RCh

\* **@return** string

\*/

**public** String country(**int** RRN, **short** id, String code, String name,

String continent, **int** area, **long** population, **float** life) {

DecimalFormat noFormat = **new** DecimalFormat("#000");

**return** String.*format*("[%s] %s ", noFormat.format(RRN),

noFormat.format(id))

+ toFormat(code, name, continent, area, population, life);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Preformatted string for printing country data.

\* **@param** code

\* **@param** name

\* **@param** continent

\* **@param** area

\* **@param** population

\* **@param** life

\* **@return** string

\*/

**public** String toFormat(String code, String name, String continent, **int** area,

**long** population, **float** life) {

**if** (name.length() > 18)

name = name.substring(0, 18);

DecimalFormat noFormat = **new** DecimalFormat("#,###");

**return** code

+ " "

+ String.*format*("%-16s %-13s %10s %13s %4.1f", name, continent,

noFormat.format(area),

noFormat.format(population), life);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Confirmation string.

\* **@param** action

\* **@return** string

\*/

**public** String Msg(String action){

**return** " OK, country "+action+".";

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Read records message.

\* **@param** recRead

\* **@return**

\*/

**public** String recReadMsg(**int** recRead) {

**return** " " + recRead + " data records read";

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Sorry message.

\* **@param** action

\* **@return** string

\*/

**public** String sorry() {

**return** " SORRY, no country with that id.";

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Sorry message.

\* **@param** action

\* **@return** string

\*/

**public** String sorry2() {

**return** " SORRY, another country has that id.";

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Sorrry message.

\* **@param** action

\* **@return** string

\*/

**public** String sorry3() {

**return** " SORRY, invalid id.";

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Sorry message.

\* **@param** action

\* **@return** string

\*/

**public** String dummy(String method) {

**return** " SORRY, " + method + " not yet working.";

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Header format.

\* **@return** string

\*/

**public** String header() {

**return** "[RRN] ID CDE NAME------------ CONTINENT---- ------AREA ---POPULATION

LIFE";

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Footer format.

\* **@return** string

\*/

**public** String footer() {

**return** "++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++";

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Writes string to file

\* **@param** toPrint string to write to file

\* **@throws** IOException

\*/

**public** **void** displayThis(String toPrint) **throws** IOException{

tL.println(toPrint);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Creates log entry and closes file

\* **@throws** IOException

\*/

**public** **void** finishUp() **throws** IOException{

tL.println("FILE STATUS > TheLog FILE closed.");

tL.close();

}

}

**package** edu.wmich.cs3310.asgn1;

**import** java.io.IOException;

**import** java.io.RandomAccessFile;

**import** java.text.DecimalFormat;

/\*\*

\* Manages countries by implementing a table

\* stored in a binary file.

\* Countries Of The World App 1.0

\* **@author** Caleb Viola

\*/

# **public** **class** CountryData {

**private** **short** N;

**private** **boolean**[] status;

**private** **boolean** located;

**private** RandomAccessFile file;

**private** CountryDataRec cDr;

**private** **final** **int** MAX\_N\_LOC = 30;

**private** **int** homeRRN;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Creates binary file and creates CountryDataRec object.

\* **@param** tL

\* **@param** fileName

\* **@throws** IOException

\*/

**public** CountryData(TheLog tL, String fileName) **throws** IOException{

file = **new** RandomAccessFile(fileName, "rw");

**try** {

N = file.readShort();

} **catch** (IOException e) {

N = 0;

}

tL.displayThis("FILE STATUS > CountryData FILE opened.");

cDr = **new** CountryDataRec();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Returns the homeRRN.

\* **@param** code

\* **@param** MAX\_N\_LOC

\* **@return**

\*/

**private** **int** hashFunction(**int** id, **int** MAX\_N\_LOC){

**if** (id % MAX\_N\_LOC == 0)

**return** MAX\_N\_LOC;

**else**

**return** id % MAX\_N\_LOC;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Cycle from 30th to 1st location.

\* **@param** homeRRN

\* **@return**

\*/

**private** **int** cycle (**int** homeRRN){

**if**(homeRRN == 30)

**return** 1;

**else**

**return** homeRRN + 1;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Execution for SI command which locates an

element in a binary file using country id.

\* **@param** Element id to locate

\* **@param** tl TheLog object

\* **@throws** IOException

\*/

**public** **void** selectByID(**int** id, TheLog tl, **boolean** fromSelectByCode)

**throws** IOException {

**if** (id != -1){

**if** (!fromSelectByCode)

homeRRN = hashFunction(id, MAX\_N\_LOC);

**else**

homeRRN = id;

**int** recRead = 0;

**int** firstVal = homeRRN;

located = **false**;

**while** (!located){

status = cDr.byteOffsetAndStatus(homeRRN, file);

**if** (status[0] && !status[1]){

cDr.readCountry(file);

**if** (cDr.getID() != id && !fromSelectByCode){

homeRRN = cycle(homeRRN);

recRead++;

**if** (homeRRN == firstVal){

tl.displayThis(tl.sorry());

tl.displayThis(tl.recReadMsg(recRead));

located = **true**;

}

}

**else**{

tl.displayThis(" " + cDr.getID()+ " "

+ tl.toFormat(cDr.getCode(), cDr.getName(),cDr.getContinent(), cDr.getArea(),cDr.getPopulation(),

cDr.getLifeExpectancy()));

recRead++;

located = **true**;

tl.displayThis(tl.recReadMsg(recRead));

}

}

**else** **if** ((status[0] && status[1]) || (!status[0] && status[1])){

tl.displayThis(tl.sorry());

located = **true**;

}

**else** {

tl.displayThis(tl.sorry3());

located = **true**;

}

}

}

**else**

tl.displayThis(tl.sorry());

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Method used by setup to insert countries in index.

\* **@param** id

\* **@param** code

\* **@param** name

\* **@param** continent

\* **@param** area

\* **@param** population

\* **@param** lifeExpectancy

\* **@param** tL TheLog object

\* **@param** userApp boolean to determine if confirmation message should be logged

\* **@return**

\* **@throws** IOException

\*/

**public** **int** insertCountry (**short** id, String code, String name,

String continent, **int** area, **long** population, **float** lifeExpectancy,

TheLog tL, **boolean** userApp) **throws** IOException {

located = **false**;

homeRRN = hashFunction(id, MAX\_N\_LOC);

**while** (!located){

status = cDr.byteOffsetAndStatus(homeRRN, file);

**if** ((status[0] && status[1]) || (!status[0] && status[1])){

cDr.writeCountry(file, id, code, name, continent, area,

population, lifeExpectancy);

located = **true**;

N++;

**if** (userApp)

tL.displayThis(tL.Msg("inserted"));

}

**else** **if** (status[0] && !status[1]){

homeRRN = cycle(homeRRN);

}

**else**

tL.displayThis(tL.sorry3());

}

**return** homeRRN;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Deletes country in table by id.

\* **@param** id

\* **@param** tL

\* **@throws** IOException

\*/

**public** **void** deleteByID (**int** id, TheLog tL) **throws** IOException{

tL.displayThis(" SORRY, not yet working");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Prints all countries.

\* **@param** tL TheLog object

\* **@throws** IOException

\*/

**private** **void** snapshot(TheLog tL) **throws** IOException{

tL.displayThis("CODE STATUS > Snapshot CountryData started.");

DecimalFormat noFormat = **new** DecimalFormat("#000");

tL.displayThis("N: "+N);

tL.displayThis(tL.header());

**short** rrn = 1;

status = cDr.byteOffsetAndStatus(rrn, file);

**while** (!(status[0] && status[1])){

cDr.readCountry(file);

**if**(cDr.getID() != 0)

tL.displayThis(tL.country(rrn, cDr.getID(), cDr.getCode(),

cDr.getName(), cDr.getContinent(), cDr.getArea(),

cDr.getPopulation(), cDr.getLifeExpectancy()));

**else**

tL.displayThis("["+noFormat.format(rrn)+"] "+" EMPTY");

rrn++;

status = cDr.byteOffsetAndStatus(rrn, file);

}

tL.displayThis(tL.footer());

tL.displayThis("CODE STATUS > Snapshot CountryData finished - " + N

+ " countries displayed.");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Records N in table, calls Snapshot, closes binary file.

\* **@param** tL TheLog object

\* **@param** printTable

\* **@throws** IOException

\*/

**public** **void** finishUp(TheLog tL, **boolean** printTable) **throws** IOException{

file.seek(0);

file.writeShort(N);

**if** (printTable)

snapshot(tL);

file.close();

tL.displayThis("FILE STATUS > CountryData FILE closed.");

}

}

**package** edu.wmich.cs3310.asgn1;

**import** java.io.File;

**import** java.io.IOException;

**import** java.io.PrintWriter;

**import** java.text.DecimalFormat;

**import** java.util.Arrays;

**import** java.util.Scanner;

/\*\*

\* **@author** Caleb Viola

\*/

# **public** **class** CountryIndex {

**private** **short** N;

**private** **final** **int** MAX\_N\_HOME\_LOC = 20;

**private** Node[] index;

**private** **int** homeSubscript;

**private** **int** nHome;

**private** **int** nColl;

**private** **int** nodRead;

**private** **int** nextEmpty;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* **@param** tl

\* **@throws** IOException

\*/

**public** CountryIndex(TheLog tl) **throws** IOException {

N = 0;

index = **new** Node[MAX\_N\_HOME\_LOC];

nHome = 0;

nColl = 0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Returns homeSubscript.

\* **@param** code

\* **@param** MAX\_N\_HOME\_LOC

\* **@return**

\*/

**private** **int** hashFunction(String code, **int** MAX\_N\_HOME\_LOC){

**return** ((**int**)code.charAt(0)\*(**int**)code.charAt(1)

\*(**int**)code.charAt(2))% MAX\_N\_HOME\_LOC;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Execution for SC command which locates an

element using country code.

\* **@param** Element id to locate

\* **@param** tl TheLog object

\* **@throws** IOException

\*/

**public** **int** selectByCode(String code, TheLog tl) **throws** IOException{

homeSubscript = hashFunction(code, MAX\_N\_HOME\_LOC);

nodRead = 0;

**return** checkLocation(homeSubscript, code);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Recursive method to examine collision locations.

\* **@param** homeSubscript

\* **@return**

\*/

**private** **int** checkLocation(**int** homeSubscript, String code) {

**if** (index[homeSubscript] == **null**)

**return** -1;

**else** **if** (index[homeSubscript].getLink() == -1)

**if** (index[homeSubscript].getCode().equals(code))

**return** index[homeSubscript].getDRP();

**else**

**return** -1;

**else**

**if** (index[homeSubscript].getCode().equals(code))

**return** index[homeSubscript].getDRP();

**else**{

nodRead++;

**return** checkLocation(index[homeSubscript].getLink(), code);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Returns number of visited node.

\* **@return** the nodRead

\*/

**public** **int** getNodRead() {

**return** nodRead;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Execution for IN command which inserts country in index.

\* **@param** code

\* **@param** rrn

\* **@param** tl TheLog object

\* **@param** userApp boolean to determine if confirmation message should be logged

\* **@throws** IOException

\*/

**public** **void** insertCodeInIndex(String code, **int** rrn, TheLog tl,

**boolean** userApp) **throws** IOException {

homeSubscript = hashFunction(code, MAX\_N\_HOME\_LOC);

nextEmpty = MAX\_N\_HOME\_LOC + nColl;

**if** (index[homeSubscript] != **null**){

expandArray();

index[nextEmpty] = **new** Node(code, rrn, -1);

**if** (index[homeSubscript].getLink() != -1){

index[nextEmpty].setLink(index[homeSubscript].getLink());

index[homeSubscript].setLink(nextEmpty);

}

**else**

index[homeSubscript].setLink(nextEmpty);

nColl++;

}

**else**{

index[homeSubscript] = **new** Node(code, rrn, -1);

nHome++;

}

**if** (userApp)

tl.displayThis(tl.Msg("inserted"));

// tl.displayThis(" SORRY, not yet working");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Deletes country in table by code.

\* **@param** id

\* **@param** tl

\* **@throws** IOException

\*/

**public** **void** deleteByCode(String code, TheLog tl) **throws** IOException{

tl.displayThis(" SORRY, not yet working.");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Backs up index to file.

\* **@throws** IOException

\*/

**public** **void** backupIndex(TheLog tL) **throws** IOException {

PrintWriter file = **new** PrintWriter("IndexBackup.csv");

tL.displayThis("FILE STATUS > IndexBackup File opened.");

**for** (**int** i = 0; i < index.length; i++){

**if**(index[i] != **null**)

file.println(index[i].getCode() + ","

+ index[i].getDRP() + ","

+ index[i].getLink());

**else**

file.println(",,-1");

}

file.close();

tL.displayThis("FILE STATUS > IndexBackup File backed up and closed.");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Restores index from file.

\* **@throws** IOException

\*/

**public** **void** restoreIndex(TheLog tL) **throws** IOException {

File file = **new** File ("IndexBackup.csv");

tL.displayThis("FILE STATUS > IndexBackup File opened.");

Scanner input = **new** Scanner (file);

String[] data;

**int** i = 0;

**while** (input.hasNextLine()){

data = input.nextLine().split(",");

**if** (!data[0].equals("")){

**if** (i < MAX\_N\_HOME\_LOC)

index[i] = **new** Node(data[0],

Integer.*parseInt*(data[1]),

Integer.*parseInt*(data[2]));

**else**{

expandArray();

index[i] = **new** Node(data[0],

Integer.*parseInt*(data[1]),

Integer.*parseInt*(data[2]));

}

}

i++;

}

input.close();

tL.displayThis("FILE STATUS > IndexBackup File restored and closed.");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Expands Node array by 1.

\*/

**public** **void** expandArray(){

index = Arrays.*copyOf*(index, index.length+1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Prints index file.

\* **@param** tL

\* **@throws** IOException

\*/

**private** **void** snapshot(TheLog tL) **throws** IOException {

tL.displayThis("CODE STATUS > Snapshot CountryIndex started.");

DecimalFormat noFormat;

tL.displayThis(String.*format*(

"CODE INDEX > MAX\_N\_HOME\_LOC: %d, nHome: %d, nColl: %d",

MAX\_N\_HOME\_LOC, nHome, nColl)+"\n");

tL.displayThis("[SUB] CODE | DRP | LINK |");

**for** (**int** i = 0; i < index.length; i++){

**if** (index[i] != **null**){

**if** (index[i].getLink() != -1)

noFormat = **new** DecimalFormat("000");

**else**

noFormat = **new** DecimalFormat("#00");

tL.displayThis(String.*format*("[%s] %s | %s | %s |",

**new** DecimalFormat("000").format(i),

index[i].getCode(),

**new** DecimalFormat("000").format(index[i]

.getDRP()),

noFormat.format(index[i].getLink())));

N++;

}

**else**

tL.displayThis(String.*format*("[%s] EMPTY", **new**

DecimalFormat("000").format(i)));

}

tL.displayThis("CODE STATUS > Snapshot CountryIndex finished - " + N

+ " countries displayed.");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* **@param** tL

\* **@param** b

\* **@throws** IOException

\*/

**public** **void** finishUp(TheLog tL, **boolean** printTable) **throws** IOException {

**if** (printTable)

snapshot(tL);

backupIndex(tL);

}

}

**package** edu.wmich.cs3310.asgn1;

**import** java.io.IOException;

**import** java.io.RandomAccessFile;

/\*\*

\* Intended to manage binary file handling.

\* **@author** Caleb Viola

\*/

# **public** **class** CountryDataRec {

**private** **short** id;

**private** String code ;

**private** String name;

**private** String continent;

**private** **int** area;

**private** **long** population;

**private** **float** lifeExpectancy;

**private** **int** byteOffset;

**private** **int** sizeOfHeaderRec = 2;

**private** **int** sizeOfDataRec = 2 + (2 \* 3) + (2 \* 16) + (2 \* 13) + 4 + 8 + 4;

/\*\*

\* For reading a country from record.

\* **@param** file

\* **@throws** IOException

\*/

**public** **void** readCountry(RandomAccessFile file) **throws** IOException{

code = "";

name = "";

continent = "";

id = file.readShort();

**for** (**int** i = 0; i < 3; i++)

code += file.readChar();

**for** (**int** i = 0; i < 16; i++)

name += file.readChar();

**for** (**int** i = 0; i < 13; i++)

continent += file.readChar();

area = file.readInt();

population = file.readLong();

lifeExpectancy = file.readFloat();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* For writing one country to record.

\* **@param** file

\* **@param** id

\* **@param** code

\* **@param** name

\* **@param** continent

\* **@param** area

\* **@param** population

\* **@param** lifeExpectancy

\* **@throws** IOException

\*/

**public** **void** writeCountry(RandomAccessFile file, **short** id,

String code, String name, String continent, **int** area,

**long** population, **float** lifeExpectancy) **throws** IOException {

file.writeShort(id);

**if** (code.length() < 3)

**while** (code.length() != 3)

code += " ";

**else** **if** (code.length() > 3)

code = code.substring(0, 3);

file.writeChars(code);

**if** (name.length() < 16)

**while** (name.length() != 16)

name += " ";

**else** **if** (name.length() > 16)

name = name.substring(0, 16);

file.writeChars(name);

**if** (continent.length() < 13)

**while** (continent.length() != 13)

continent += " ";

**else** **if** (continent.length() > 13)

continent = continent.substring(0, 13);

file.writeChars(continent);

file.writeInt(area);

file.writeLong(population);

file.writeFloat(lifeExpectancy);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* For calculating and locating byteOffset, and returning the status

of the file at a given address by means of booleans.

\* **@param** RRN

\* **@param** file

\* **@return** spot status.

\* **@throws** IOException

\*/

**public** **boolean**[] byteOffsetAndStatus(**int** RRN, RandomAccessFile file){

byteOffset = sizeOfHeaderRec + ((RRN - 1) \* sizeOfDataRec);

**try** {

file.seek(byteOffset);

**try** {

**int** test = file.readShort();

file.seek(byteOffset);

**if** (test > 0)

**return** **new** **boolean**[] { **true**, **false** }; /\* Existent id. \*/

**return** **new** **boolean**[] { **false**, **true** }; /\* Empty space exists

within existent

spots. \*/

} **catch** (IOException e) {

**return** **new** **boolean**[] { **true**, **true** };/\* New spot; not between

others. \*/

}

} **catch** (IOException e) {

**return** **new** **boolean**[] { **false**, **false** }; /\* Out of range. \*/

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Deletes country from the record by adding 0s.

\* **@param** file

\* **@throws** IOException

\*/

**public** **void** deleteCountry(RandomAccessFile file) **throws** IOException {

file.seek(getByteOffset());

**for** (**int** i = 0; i < sizeOfDataRec + sizeOfHeaderRec; i++){

file.writeByte(0000);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for id.

\* **@return**

\*/

**public** **short** getID(){

**return** id;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for code.

\* **@return**

\*/

**public** String getCode(){

**return** code;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for name.

\* **@return**

\*/

**public** String getName(){

**return** name;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for continent.

\* **@return**

\*/

**public** String getContinent(){

**return** continent;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for area.

\* **@return**

\*/

**public** **int** getArea(){

**return** area;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for population.

\* **@return**

\*/

**public** **long** getPopulation(){

**return** population;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for lifeExpectancy.

\* **@return**

\*/

**public** **float** getLifeExpectancy(){

**return** lifeExpectancy;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for byteOffset.

\* **@return**

\*/

**public** **int** getByteOffset() {

**return** byteOffset;

}

}

**package** edu.wmich.cs3310.asgn1;

/\*\*

\* Index node.

\* **@author** Caleb Viola

\*/

# **public** **class** Node {

**private** String code ;

**private** **int** DRP;

**private** **int** link;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Node constructor.

\* **@param** code

\* **@param** DRP

\* **@param** link

\*/

**public** Node(String code, **int** DRP, **int** link){

**this**.code = code;

**this**.DRP = DRP;

**this**.link = link;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for code.

\* **@return** the code

\*/

**public** String getCode() {

**return** code;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for DRP;

\* **@return** the dRP

\*/

**public** **int** getDRP() {

**return** DRP;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Setter for link.

\* **@param** link

\*/

**public** **void** setLink(**int** link) {

**this**.link = link;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Getter for link.

\* **@return** the link

\*/

**public** **int** getLink() {

**return** link;

}

}