Countries Of The World App 1.0 - Java

CS 3310 – Dr. Donna Kaminski

Index

[Log.txt 2](#_Toc400386338)

[CountryData.txt 10](#_Toc400386339)

[Backup.txt 11](#_Toc400386340)

[TestDriver 12](#_Toc400386341)

[Setup 13](#_Toc400386342)

[UserApp 14](#_Toc400386343)

[PrettyPrintUtility 16](#_Toc400386344)

[RawData 17](#_Toc400386345)

[UI 19](#_Toc400386346)

[DataTable 22](#_Toc400386347)

[DataTableRecord 25](#_Toc400386348)

[NameIndex 28](#_Toc400386349)

[BSTNode 31](#_Toc400386350)

[Log 32](#_Toc400386351)

[Backup 34](#_Toc400386352)

# FILE STATUS > Log FILE opened

CODE STATUS > Setup started

FILE STATUS > RawData FILE opened (A1RawDataSample.csv)

FILE STATUS > CountryData FILE opened

FILE STATUS > RawData FILE closed

CODE STATUS > Setup finished - 25 countries processed

FILE STATUS > Log FILE closed

CODE STATUS > PrettyPrintUtility started

N: 25 | MaxId: 35

DATA STORAGE

LOC/ CDE ID- NAME-------------- CONTINENT---- ------AREA ---POPULATION LIFE

001/ GRC 001 Greece Europe 131,626 10,545,700 78.4

002/ EGY 002 Egypt Africa 1,001,449 68,470,000 63.3

003/ 003 ...

004/ 004 ...

005/ CHN 005 China Asia 9,572,900 1,277,558,000 71.4

006/ 006 ...

007/ AUS 007 Australia Oceania 7,741,220 18,886,000 79.8

008/ USA 008 United States North America 9,363,520 278,357,000 77.1

009/ YUG 009 Yugoslavia Europe 102,173 10,640,000 72.4

010/ BRA 010 Brazil South America 8,547,404 170,115,000 62.9

011/ TUR 011 Turkey Asia 774,815 66,591,000 71.0

012/ 012 ...

013/ PSE 013 Palestine Asia 6,257 3,101,000 71.4

014/ MEX 014 Mexico North America 1,958,201 98,881,000 71.5

015/ HUN 015 Hungary Europe 93,030 10,043,200 71.4

016/ ZWE 016 Zimbabwe Africa 390,757 11,669,000 37.8

017/ WLF 017 Wallis and Futu Oceania 200 15,000 0.0

018/ 018 ...

019/ FRA 019 France Europe 551,500 59,225,700 78.8

020/ 020 ...

021/ QAT 021 Qatar Asia 11,000 599,000 72.4

022/ OMN 022 Oman Asia 309,500 2,542,000 71.8

023/ 023 ...

024/ DOM 024 Dominican Repub North America 48,511 8,495,000 73.2

025/ NGA 025 Nigeria Africa 923,768 111,506,000 51.6

026/ JOR 026 Jordan Asia 88,946 5,083,000 77.4

027/ 027 ...

028/ KEN 028 Kenya Africa 580,367 30,080,000 48.0

029/ VEN 029 Venezuela South America 912,050 24,170,000 73.1

030/ RUS 030 Russian Federat Europe 17,075,400 146,934,000 67.2

031/ SWE 031 Sweden Europe 449,964 8,861,400 79.6

032/ 032 ...

033/ 033 ...

034/ LIE 034 Liechtenstein Europe 160 32,300 78.8

035/ IND 035 India Asia 3,287,263 1,013,662,000 62.5

NAME INDEX

N: 25 | NextEmpty: 25 | RootPtr: 0

LOC/ Lch NAME-------------- PTR Rch

000/ 003 China 005 001

001/ 007 India 035 002

002/ 004 United States 008 011

003/ 012 Brazil 010 -01

004/ 005 Russian Federation 030 008

005/ 006 Nigeria 025 020

006/ 010 Mexico 014 -01

007/ 018 Egypt 002 009

008/ 017 Turkey 011 -01

009/ -01 France 019 015

010/ 019 Kenya 028 023

011/ -01 Venezuela 029 013

012/ -01 Australia 007 -01

013/ 014 Zimbabwe 016 -01

014/ 024 Yugoslavia 009 -01

015/ -01 Greece 001 016

016/ -01 Hungary 015 -01

017/ -01 Sweden 031 -01

018/ -01 Dominican Republic 024 -01

019/ -01 Jordan 026 -01

020/ 021 Palestine 013 022

021/ -01 Oman 022 -01

022/ -01 Qatar 021 -01

023/ -01 Liechtenstein 034 -01

024/ -01 Wallis and Futuna 017 -01

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

CODE STATUS > PrettyPrintUtility finished

FILE STATUS > Log FILE opened

CODE STATUS > UserApp started

FILE STATUS > TransData FILE opened (A1TransData1.txt)

FILE STATUS > CountryData FILE opened

AN

CDE ID- NAME-------------- CONTINENT---- ------AREA ---POPULATION LIFE

AUS 007 Australia Oceania 7,741,220 18,886,000 79.8

BRA 010 Brazil South America 8,547,404 170,115,000 62.9

CHN 005 China Asia 9,572,900 1,277,558,000 71.4

DOM 024 Dominican Repub North America 48,511 8,495,000 73.2

EGY 002 Egypt Africa 1,001,449 68,470,000 63.3

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

GRC 001 Greece Europe 131,626 10,545,700 78.4

HUN 015 Hungary Europe 93,030 10,043,200 71.4

IND 035 India Asia 3,287,263 1,013,662,000 62.5

JOR 026 Jordan Asia 88,946 5,083,000 77.4

KEN 028 Kenya Africa 580,367 30,080,000 48.0

LIE 034 Liechtenstein Europe 160 32,300 78.8

MEX 014 Mexico North America 1,958,201 98,881,000 71.5

NGA 025 Nigeria Africa 923,768 111,506,000 51.6

OMN 022 Oman Asia 309,500 2,542,000 71.8

PSE 013 Palestine Asia 6,257 3,101,000 71.4

QAT 021 Qatar Asia 11,000 599,000 72.4

RUS 030 Russian Federat Europe 17,075,400 146,934,000 67.2

SWE 031 Sweden Europe 449,964 8,861,400 79.6

TUR 011 Turkey Asia 774,815 66,591,000 71.0

USA 008 United States North America 9,363,520 278,357,000 77.1

VEN 029 Venezuela South America 912,050 24,170,000 73.1

WLF 017 Wallis and Futu Oceania 200 15,000 0.0

YUG 009 Yugoslavia Europe 102,173 10,640,000 72.4

ZWE 016 Zimbabwe Africa 390,757 11,669,000 37.8

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

DN China

[SORRY, Delete By Name module not yet working]

DN Australia

[SORRY, Delete By Name module not yet working]

SN China

CHN 005 China Asia 9,572,900 1,277,558,000 71.4

>> 1 node visited

SN Wallis

ERROR, invalid country name

>> 8 nodes visited

SN Wallis and Futuna

WLF 017 Wallis and Futu Oceania 200 15,000 0.0

>> 7 nodes visited

SN Wallis & Futuna

ERROR, invalid country name

>> 8 nodes visited

SN Australia

AUS 007 Australia Oceania 7,741,220 18,886,000 79.8

>> 3 nodes visited

SN Zimbabwe

ZWE 016 Zimbabwe Africa 390,757 11,669,000 37.8

>> 5 nodes visited

SN Jordan

JOR 026 Jordan Asia 88,946 5,083,000 77.4

>> 8 nodes visited

SN Japan

ERROR, invalid country name

>> 9 nodes visited

SN Liechtenstein

LIE 034 Liechtenstein Europe 160 32,300 78.8

>> 8 nodes visited

SN London

ERROR, invalid country name

>> 9 nodes visited

SN Albania

ERROR, invalid country name

>> 4 nodes visited

SN Western Michigan University

ERROR, invalid country name

>> 8 nodes visited

SN Zoo

ERROR, invalid country name

>> 6 nodes visited

SN United

ERROR, invalid country name

>> 6 nodes visited

SN Russian Federation

RUS 030 Russian Federat Europe 17,075,400 146,934,000 67.2

>> 4 nodes visited

SN Russian Federat

ERROR, invalid country name

>> 8 nodes visited

SN Russian Federat of Mickey Mouse

ERROR, invalid country name

>> 8 nodes visited

SN (space) is < 0-9 (& so < A=Z)

ERROR, invalid country name

>> 4 nodes visited

SN # is < 0-9 (& so < A-Z)

ERROR, invalid country name

>> 4 nodes visited

SN = is > 0-9 & < A-Z

ERROR, invalid country name

>> 4 nodes visited

SN ] is > A-Z & < a-z

ERROR, invalid country name

>> 6 nodes visited

SN } is > a-z

ERROR, invalid country name

>> 6 nodes visited

FILE STATUS > TransData FILE closed

CODE STATUS > UserApp finished - 25 transactions processed

FILE STATUS > Log FILE closed

FILE STATUS > Log FILE opened

CODE STATUS > UserApp started

FILE STATUS > TransData FILE opened (A1TransData2.txt)

FILE STATUS > CountryData FILE opened

AI

CDE ID- NAME-------------- CONTINENT---- ------AREA ---POPULATION LIFE

GRC 001 Greece Europe 131,626 10,545,700 78.4

EGY 002 Egypt Africa 1,001,449 68,470,000 63.3

CHN 005 China Asia 9,572,900 1,277,558,000 71.4

AUS 007 Australia Oceania 7,741,220 18,886,000 79.8

USA 008 United States North America 9,363,520 278,357,000 77.1

YUG 009 Yugoslavia Europe 102,173 10,640,000 72.4

BRA 010 Brazil South America 8,547,404 170,115,000 62.9

TUR 011 Turkey Asia 774,815 66,591,000 71.0

PSE 013 Palestine Asia 6,257 3,101,000 71.4

MEX 014 Mexico North America 1,958,201 98,881,000 71.5

HUN 015 Hungary Europe 93,030 10,043,200 71.4

ZWE 016 Zimbabwe Africa 390,757 11,669,000 37.8

WLF 017 Wallis and Futu Oceania 200 15,000 0.0

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

QAT 021 Qatar Asia 11,000 599,000 72.4

OMN 022 Oman Asia 309,500 2,542,000 71.8

DOM 024 Dominican Repub North America 48,511 8,495,000 73.2

NGA 025 Nigeria Africa 923,768 111,506,000 51.6

JOR 026 Jordan Asia 88,946 5,083,000 77.4

KEN 028 Kenya Africa 580,367 30,080,000 48.0

VEN 029 Venezuela South America 912,050 24,170,000 73.1

RUS 030 Russian Federat Europe 17,075,400 146,934,000 67.2

SWE 031 Sweden Europe 449,964 8,861,400 79.6

LIE 034 Liechtenstein Europe 160 32,300 78.8

IND 035 India Asia 3,287,263 1,013,662,000 62.5

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

DI 1

[SORRY, Delete By Id module not yet working]

DI 25

[SORRY, Delete By Id module not yet working]

SI 35

IND 035 India Asia 3,287,263 1,013,662,000 62.5

SI 17

WLF 017 Wallis and Futu Oceania 200 15,000 0.0

SI 999999999

SORRY, no country with that id

SI 0

SORRY, no country with that id

SI 1

GRC 001 Greece Europe 131,626 10,545,700 78.4

SI 5

CHN 005 China Asia 9,572,900 1,277,558,000 71.4

SI 20

SORRY, no country with that id

SI 3

SORRY, no country with that id

SI 19

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

SI 20

SORRY, no country with that id

AI

CDE ID- NAME-------------- CONTINENT---- ------AREA ---POPULATION LIFE

GRC 001 Greece Europe 131,626 10,545,700 78.4

EGY 002 Egypt Africa 1,001,449 68,470,000 63.3

CHN 005 China Asia 9,572,900 1,277,558,000 71.4

AUS 007 Australia Oceania 7,741,220 18,886,000 79.8

USA 008 United States North America 9,363,520 278,357,000 77.1

YUG 009 Yugoslavia Europe 102,173 10,640,000 72.4

BRA 010 Brazil South America 8,547,404 170,115,000 62.9

TUR 011 Turkey Asia 774,815 66,591,000 71.0

PSE 013 Palestine Asia 6,257 3,101,000 71.4

MEX 014 Mexico North America 1,958,201 98,881,000 71.5

HUN 015 Hungary Europe 93,030 10,043,200 71.4

ZWE 016 Zimbabwe Africa 390,757 11,669,000 37.8

WLF 017 Wallis and Futu Oceania 200 15,000 0.0

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

QAT 021 Qatar Asia 11,000 599,000 72.4

OMN 022 Oman Asia 309,500 2,542,000 71.8

DOM 024 Dominican Repub North America 48,511 8,495,000 73.2

NGA 025 Nigeria Africa 923,768 111,506,000 51.6

JOR 026 Jordan Asia 88,946 5,083,000 77.4

KEN 028 Kenya Africa 580,367 30,080,000 48.0

VEN 029 Venezuela South America 912,050 24,170,000 73.1

RUS 030 Russian Federat Europe 17,075,400 146,934,000 67.2

SWE 031 Sweden Europe 449,964 8,861,400 79.6

LIE 034 Liechtenstein Europe 160 32,300 78.8

IND 035 India Asia 3,287,263 1,013,662,000 62.5

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

FILE STATUS > TransData FILE closed

CODE STATUS > UserApp finished - 14 transactions processed

FILE STATUS > Log FILE closed

FILE STATUS > Log FILE opened

CODE STATUS > UserApp started

FILE STATUS > TransData FILE opened (A1TransData3.txt)

FILE STATUS > CountryData FILE opened

AI

CDE ID- NAME-------------- CONTINENT---- ------AREA ---POPULATION LIFE

GRC 001 Greece Europe 131,626 10,545,700 78.4

EGY 002 Egypt Africa 1,001,449 68,470,000 63.3

CHN 005 China Asia 9,572,900 1,277,558,000 71.4

AUS 007 Australia Oceania 7,741,220 18,886,000 79.8

USA 008 United States North America 9,363,520 278,357,000 77.1

YUG 009 Yugoslavia Europe 102,173 10,640,000 72.4

BRA 010 Brazil South America 8,547,404 170,115,000 62.9

TUR 011 Turkey Asia 774,815 66,591,000 71.0

PSE 013 Palestine Asia 6,257 3,101,000 71.4

MEX 014 Mexico North America 1,958,201 98,881,000 71.5

HUN 015 Hungary Europe 93,030 10,043,200 71.4

ZWE 016 Zimbabwe Africa 390,757 11,669,000 37.8

WLF 017 Wallis and Futu Oceania 200 15,000 0.0

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

QAT 021 Qatar Asia 11,000 599,000 72.4

OMN 022 Oman Asia 309,500 2,542,000 71.8

DOM 024 Dominican Repub North America 48,511 8,495,000 73.2

NGA 025 Nigeria Africa 923,768 111,506,000 51.6

JOR 026 Jordan Asia 88,946 5,083,000 77.4

KEN 028 Kenya Africa 580,367 30,080,000 48.0

VEN 029 Venezuela South America 912,050 24,170,000 73.1

RUS 030 Russian Federat Europe 17,075,400 146,934,000 67.2

SWE 031 Sweden Europe 449,964 8,861,400 79.6

LIE 034 Liechtenstein Europe 160 32,300 78.8

IND 035 India Asia 3,287,263 1,013,662,000 62.5

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

AN

CDE ID- NAME-------------- CONTINENT---- ------AREA ---POPULATION LIFE

AUS 007 Australia Oceania 7,741,220 18,886,000 79.8

BRA 010 Brazil South America 8,547,404 170,115,000 62.9

CHN 005 China Asia 9,572,900 1,277,558,000 71.4

DOM 024 Dominican Repub North America 48,511 8,495,000 73.2

EGY 002 Egypt Africa 1,001,449 68,470,000 63.3

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

GRC 001 Greece Europe 131,626 10,545,700 78.4

HUN 015 Hungary Europe 93,030 10,043,200 71.4

IND 035 India Asia 3,287,263 1,013,662,000 62.5

JOR 026 Jordan Asia 88,946 5,083,000 77.4

KEN 028 Kenya Africa 580,367 30,080,000 48.0

LIE 034 Liechtenstein Europe 160 32,300 78.8

MEX 014 Mexico North America 1,958,201 98,881,000 71.5

NGA 025 Nigeria Africa 923,768 111,506,000 51.6

OMN 022 Oman Asia 309,500 2,542,000 71.8

PSE 013 Palestine Asia 6,257 3,101,000 71.4

QAT 021 Qatar Asia 11,000 599,000 72.4

RUS 030 Russian Federat Europe 17,075,400 146,934,000 67.2

SWE 031 Sweden Europe 449,964 8,861,400 79.6

TUR 011 Turkey Asia 774,815 66,591,000 71.0

USA 008 United States North America 9,363,520 278,357,000 77.1

VEN 029 Venezuela South America 912,050 24,170,000 73.1

WLF 017 Wallis and Futu Oceania 200 15,000 0.0

YUG 009 Yugoslavia Europe 102,173 10,640,000 72.4

ZWE 016 Zimbabwe Africa 390,757 11,669,000 37.8

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

SI 31

SWE 031 Sweden Europe 449,964 8,861,400 79.6

SN Sweden

SWE 031 Sweden Europe 449,964 8,861,400 79.6

>> 6 nodes visited

SN Wallis and Futuna

WLF 017 Wallis and Futu Oceania 200 15,000 0.0

>> 7 nodes visited

SI 17

WLF 017 Wallis and Futu Oceania 200 15,000 0.0

IN ARG,27,Argentina,2780400,37032000,75.1

OK, country inserted (in data storage & name index)

>> 3 nodes visited in the name index

IN JAM,12,Jamaica,10990,2583000,75.2

OK, country inserted (in data storage & name index)

>> 8 nodes visited in the name index

IN CUB,20,Cuba,110861,11201000,76.2

OK, country inserted (in data storage & name index)

>> 4 nodes visited in the name index

IN AUT,39,Austria,83859,8091800,77.7

OK, country inserted (in data storage & name index)

>> 3 nodes visited in the name index

IN BWA,41,Botswana,581730,1622000,39.3

OK, country inserted (in data storage & name index)

>> 4 nodes visited in the name index

IN KAZ,6,Kazakstan,2724900,16223000,63.2

OK, country inserted (in data storage & name index)

>> 8 nodes visited in the name index

SN Kazakstan

KAZ 006 Kazakstan Asia 2,724,900 16,223,000 63.2

>> 9 nodes visited

SI 27

ARG 027 Argentina South America 2,780,400 37,032,000 75.1

SI 6

KAZ 006 Kazakstan Asia 2,724,900 16,223,000 63.2

SN AUstria

ERROR, invalid country name

>> 5 nodes visited

SN Botswana

BWA 041 Botswana Africa 581,730 1,622,000 39.3

>> 5 nodes visited

SI 39

AUT 039 Austria Europe 83,859 8,091,800 77.7

SI 12

JAM 012 Jamaica North America 10,990 2,583,000 75.2

SN Argentina

ARG 027 Argentina South America 2,780,400 37,032,000 75.1

>> 4 nodes visited

SN Jamaica

JAM 012 Jamaica North America 10,990 2,583,000 75.2

>> 9 nodes visited

SI 41

BWA 041 Botswana Africa 581,730 1,622,000 39.3

SI 20

CUB 020 Cuba North America 110,861 11,201,000 76.2

SN Cuba

CUB 020 Cuba North America 110,861 11,201,000 76.2

>> 5 nodes visited

SI 40

SORRY, no country with that id

SI 999999999

SORRY, no country with that id

SI 42

SORRY, no country with that id

AI

CDE ID- NAME-------------- CONTINENT---- ------AREA ---POPULATION LIFE

GRC 001 Greece Europe 131,626 10,545,700 78.4

EGY 002 Egypt Africa 1,001,449 68,470,000 63.3

CHN 005 China Asia 9,572,900 1,277,558,000 71.4

KAZ 006 Kazakstan Asia 2,724,900 16,223,000 63.2

AUS 007 Australia Oceania 7,741,220 18,886,000 79.8

USA 008 United States North America 9,363,520 278,357,000 77.1

YUG 009 Yugoslavia Europe 102,173 10,640,000 72.4

BRA 010 Brazil South America 8,547,404 170,115,000 62.9

TUR 011 Turkey Asia 774,815 66,591,000 71.0

JAM 012 Jamaica North America 10,990 2,583,000 75.2

PSE 013 Palestine Asia 6,257 3,101,000 71.4

MEX 014 Mexico North America 1,958,201 98,881,000 71.5

HUN 015 Hungary Europe 93,030 10,043,200 71.4

ZWE 016 Zimbabwe Africa 390,757 11,669,000 37.8

WLF 017 Wallis and Futu Oceania 200 15,000 0.0

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

CUB 020 Cuba North America 110,861 11,201,000 76.2

QAT 021 Qatar Asia 11,000 599,000 72.4

OMN 022 Oman Asia 309,500 2,542,000 71.8

DOM 024 Dominican Repub North America 48,511 8,495,000 73.2

NGA 025 Nigeria Africa 923,768 111,506,000 51.6

JOR 026 Jordan Asia 88,946 5,083,000 77.4

ARG 027 Argentina South America 2,780,400 37,032,000 75.1

KEN 028 Kenya Africa 580,367 30,080,000 48.0

VEN 029 Venezuela South America 912,050 24,170,000 73.1

RUS 030 Russian Federat Europe 17,075,400 146,934,000 67.2

SWE 031 Sweden Europe 449,964 8,861,400 79.6

LIE 034 Liechtenstein Europe 160 32,300 78.8

IND 035 India Asia 3,287,263 1,013,662,000 62.5

AUT 039 Austria Europe 83,859 8,091,800 77.7

BWA 041 Botswana Africa 581,730 1,622,000 39.3

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

AN

CDE ID- NAME-------------- CONTINENT---- ------AREA ---POPULATION LIFE

ARG 027 Argentina South America 2,780,400 37,032,000 75.1

AUS 007 Australia Oceania 7,741,220 18,886,000 79.8

AUT 039 Austria Europe 83,859 8,091,800 77.7

BWA 041 Botswana Africa 581,730 1,622,000 39.3

BRA 010 Brazil South America 8,547,404 170,115,000 62.9

CHN 005 China Asia 9,572,900 1,277,558,000 71.4

CUB 020 Cuba North America 110,861 11,201,000 76.2

DOM 024 Dominican Repub North America 48,511 8,495,000 73.2

EGY 002 Egypt Africa 1,001,449 68,470,000 63.3

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

GRC 001 Greece Europe 131,626 10,545,700 78.4

HUN 015 Hungary Europe 93,030 10,043,200 71.4

IND 035 India Asia 3,287,263 1,013,662,000 62.5

JAM 012 Jamaica North America 10,990 2,583,000 75.2

JOR 026 Jordan Asia 88,946 5,083,000 77.4

KAZ 006 Kazakstan Asia 2,724,900 16,223,000 63.2

KEN 028 Kenya Africa 580,367 30,080,000 48.0

LIE 034 Liechtenstein Europe 160 32,300 78.8

MEX 014 Mexico North America 1,958,201 98,881,000 71.5

NGA 025 Nigeria Africa 923,768 111,506,000 51.6

OMN 022 Oman Asia 309,500 2,542,000 71.8

PSE 013 Palestine Asia 6,257 3,101,000 71.4

QAT 021 Qatar Asia 11,000 599,000 72.4

RUS 030 Russian Federat Europe 17,075,400 146,934,000 67.2

SWE 031 Sweden Europe 449,964 8,861,400 79.6

TUR 011 Turkey Asia 774,815 66,591,000 71.0

USA 008 United States North America 9,363,520 278,357,000 77.1

VEN 029 Venezuela South America 912,050 24,170,000 73.1

WLF 017 Wallis and Futu Oceania 200 15,000 0.0

YUG 009 Yugoslavia Europe 102,173 10,640,000 72.4

ZWE 016 Zimbabwe Africa 390,757 11,669,000 37.8

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

FILE STATUS > TransData FILE closed

CODE STATUS > UserApp finished - 29 transactions processed

FILE STATUS > Log FILE closed

CODE STATUS > PrettyPrintUtility started

N: 31 | MaxId: 41

DATA STORAGE

LOC/ CDE ID- NAME-------------- CONTINENT---- ------AREA ---POPULATION LIFE

001/ GRC 001 Greece Europe 131,626 10,545,700 78.4

002/ EGY 002 Egypt Africa 1,001,449 68,470,000 63.3

003/ 003 ...

004/ 004 ...

005/ CHN 005 China Asia 9,572,900 1,277,558,000 71.4

006/ KAZ 006 Kazakstan Asia 2,724,900 16,223,000 63.2

007/ AUS 007 Australia Oceania 7,741,220 18,886,000 79.8

008/ USA 008 United States North America 9,363,520 278,357,000 77.1

009/ YUG 009 Yugoslavia Europe 102,173 10,640,000 72.4

010/ BRA 010 Brazil South America 8,547,404 170,115,000 62.9

011/ TUR 011 Turkey Asia 774,815 66,591,000 71.0

012/ JAM 012 Jamaica North America 10,990 2,583,000 75.2

013/ PSE 013 Palestine Asia 6,257 3,101,000 71.4

014/ MEX 014 Mexico North America 1,958,201 98,881,000 71.5

015/ HUN 015 Hungary Europe 93,030 10,043,200 71.4

016/ ZWE 016 Zimbabwe Africa 390,757 11,669,000 37.8

017/ WLF 017 Wallis and Futu Oceania 200 15,000 0.0

018/ 018 ...

019/ FRA 019 France Europe 551,500 59,225,700 78.8

020/ CUB 020 Cuba North America 110,861 11,201,000 76.2

021/ QAT 021 Qatar Asia 11,000 599,000 72.4

022/ OMN 022 Oman Asia 309,500 2,542,000 71.8

023/ 023 ...

024/ DOM 024 Dominican Repub North America 48,511 8,495,000 73.2

025/ NGA 025 Nigeria Africa 923,768 111,506,000 51.6

026/ JOR 026 Jordan Asia 88,946 5,083,000 77.4

027/ ARG 027 Argentina South America 2,780,400 37,032,000 75.1

028/ KEN 028 Kenya Africa 580,367 30,080,000 48.0

029/ VEN 029 Venezuela South America 912,050 24,170,000 73.1

030/ RUS 030 Russian Federat Europe 17,075,400 146,934,000 67.2

031/ SWE 031 Sweden Europe 449,964 8,861,400 79.6

032/ 032 ...

033/ 033 ...

034/ LIE 034 Liechtenstein Europe 160 32,300 78.8

035/ IND 035 India Asia 3,287,263 1,013,662,000 62.5

036/ 036 ...

037/ 037 ...

038/ 038 ...

039/ AUT 039 Austria Europe 83,859 8,091,800 77.7

040/ 040 ...

041/ BWA 041 Botswana Africa 581,730 1,622,000 39.3

NAME INDEX

N: 31 | NextEmpty: 31 | RootPtr: 0

LOC/ Lch NAME-------------- PTR Rch

000/ 003 China 005 001

001/ 007 India 035 002

002/ 004 United States 008 011

003/ 012 Brazil 010 -01

004/ 005 Russian Federation 030 008

005/ 006 Nigeria 025 020

006/ 010 Mexico 014 -01

007/ 018 Egypt 002 009

008/ 017 Turkey 011 -01

009/ -01 France 019 015

010/ 019 Kenya 028 023

011/ -01 Venezuela 029 013

012/ 025 Australia 007 028

013/ 014 Zimbabwe 016 -01

014/ 024 Yugoslavia 009 -01

015/ -01 Greece 001 016

016/ -01 Hungary 015 -01

017/ -01 Sweden 031 -01

018/ 027 Dominican Republic 024 -01

019/ 026 Jordan 026 030

020/ 021 Palestine 013 022

021/ -01 Oman 022 -01

022/ -01 Qatar 021 -01

023/ -01 Liechtenstein 034 -01

024/ -01 Wallis and Futuna 017 -01

025/ -01 Argentina 027 -01

026/ -01 Jamaica 012 -01

027/ -01 Cuba 020 -01

028/ -01 Austria 039 029

029/ -01 Botswana 041 -01

030/ -01 Kazakstan 006 -01

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

CODE STATUS > PrettyPrintUtility finished

# 031'041

GRC'001'Greece 'Europe '00131626'0010545700'78.4

EGY'002'Egypt 'Africa '01001449'0068470000'63.3

'000' ' '00000000'0000000000'00.0

'000' ' '00000000'0000000000'00.0

CHN'005'China 'Asia '09572900'1277558000'71.4

KAZ'006'Kazakstan 'Asia '02724900'0016223000'63.2

AUS'007'Australia 'Oceania '07741220'0018886000'79.8

USA'008'United States 'North America'09363520'0278357000'77.1

YUG'009'Yugoslavia 'Europe '00102173'0010640000'72.4

BRA'010'Brazil 'South America'08547404'0170115000'62.9

TUR'011'Turkey 'Asia '00774815'0066591000'71.0

JAM'012'Jamaica 'North America'00010990'0002583000'75.2

PSE'013'Palestine 'Asia '00006257'0003101000'71.4

MEX'014'Mexico 'North America'01958201'0098881000'71.5

HUN'015'Hungary 'Europe '00093030'0010043200'71.4

ZWE'016'Zimbabwe 'Africa '00390757'0011669000'37.8

WLF'017'Wallis and Futu'Oceania '00000200'0000015000' 0.0

'000' ' '00000000'0000000000'00.0

FRA'019'France 'Europe '00551500'0059225700'78.8

CUB'020'Cuba 'North America'00110861'0011201000'76.2

QAT'021'Qatar 'Asia '00011000'0000599000'72.4

OMN'022'Oman 'Asia '00309500'0002542000'71.8

'000' ' '00000000'0000000000'00.0

DOM'024'Dominican Repub'North America'00048511'0008495000'73.2

NGA'025'Nigeria 'Africa '00923768'0111506000'51.6

JOR'026'Jordan 'Asia '00088946'0005083000'77.4

ARG'027'Argentina 'South America'02780400'0037032000'75.1

KEN'028'Kenya 'Africa '00580367'0030080000'48.0

VEN'029'Venezuela 'South America'00912050'0024170000'73.1

RUS'030'Russian Federat'Europe '17075400'0146934000'67.2

SWE'031'Sweden 'Europe '00449964'0008861400'79.6

'000' ' '00000000'0000000000'00.0

'000' ' '00000000'0000000000'00.0

LIE'034'Liechtenstein 'Europe '00000160'0000032300'78.8

IND'035'India 'Asia '03287263'1013662000'62.5

'000' ' '00000000'0000000000'00.0

'000' ' '00000000'0000000000'00.0

'000' ' '00000000'0000000000'00.0

AUT'039'Austria 'Europe '00083859'0008091800'77.7

'000' ' '00000000'0000000000'00.0

BWA'041'Botswana 'Africa '00581730'0001622000'39.3

# 31'31'0

3'China'5'1

7'India'35'2

4'United States'8'11

12'Brazil'10'-1

5'Russian Federation'30'8

6'Nigeria'25'20

10'Mexico'14'-1

18'Egypt'2'9

17'Turkey'11'-1

-1'France'19'15

19'Kenya'28'23

-1'Venezuela'29'13

25'Australia'7'28

14'Zimbabwe'16'-1

24'Yugoslavia'9'-1

-1'Greece'1'16

-1'Hungary'15'-1

-1'Sweden'31'-1

27'Dominican Republic'24'-1

26'Jordan'26'30

21'Palestine'13'22

-1'Oman'22'-1

-1'Qatar'21'-1

-1'Liechtenstein'34'-1

-1'Wallis and Futuna'17'-1

-1'Argentina'27'-1

-1'Jamaica'12'-1

-1'Cuba'20'-1

-1'Austria'39'29

-1'Botswana'41'-1

-1'Kazakstan'6'-1

# **package** edu.wmich.cs3310.a2;

**import** java.io.File;

**import** java.io.IOException;

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

\* Creates and manages a country information table

\* Countries Of The World App 1.0

\* **@author** Caleb Viola

\*/

**public** **class** TestDriver {

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

\* According to A2DemoSpecs

\* **@param** args

\* **@throws** IOException

\*/

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

File logFile = **new** File("Log.txt");

File countryData = **new** File("CountryData.txt");

File backup = **new** File("Backup.txt");

**if**(logFile.exists()) logFile.delete();

**if**(countryData.exists()) countryData.delete();

**if**(backup.exists()) backup.delete();

Setup.*main*("Sample");

PrettyPrintUtility.*main*();

**for** (**short** i = 1; i <= 3; i++)

UserApp.*main*(i);

PrettyPrintUtility.*main*();

}

}

# **package** edu.wmich.cs3310.a2;

**import** java.io.IOException;

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

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

\* Countries Of The World App 1.0

\* **@author** Caleb Viola

\*/

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

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

\* Setup DataTable and NameIndex from RawData files

\* **@param** fileNameSufix

\* **@throws** IOException

\*/

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

Log log = **new** Log(**true**);

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

RawData rd = **new** RawData(fileNameSufix, log);

DataTable dt = **new** DataTable("CountryData.txt", log);

NameIndex ni = **new** NameIndex();

**int** N = 0;

rd.grabCountry();

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

**if**(rd.countryGrabbed){

dt.insert1Country(rd.getCode(), rd.getId(), rd.getName(),

rd.getContinent(), rd.getArea(), rd.getPopulation(),

rd.getLifeExpectancy(), log);

ni.insertIntoNameIndex(rd.getName(), rd.getId());

}

rd.grabCountry();

}

N = rd.getTransactions();

rd.finishUp(log);

dt.finishUp();

ni.finishUp();

log.displayThis("CODE STATUS > Setup finished - "+N+" countries processed");

log.finishUp();

}

}

# **package** edu.wmich.cs3310.a2;

**import** java.io.File;

**import** java.io.FileNotFoundException;

**import** java.io.IOException;

**import** java.util.Scanner;

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

\* Interact with DataTable from TransData files

\* Countries Of The World App 1.0

\* **@author** Caleb Viola

\*/

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

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

\* **@param** fileNameSufix

\* **@throws** IOException

\*/

**public** **static** **void** **main**(**short** fileNameSufix) **throws** IOException{

Log log = **new** Log(**true**);

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

UI ui = **new** UI(fileNameSufix, log);

DataTable dt = **new** DataTable("CountryData.txt", log);

NameIndex ni = **new** NameIndex();

*load*(ni);

**int** N = 0;

ui.grabCommand();

**while** (!ui.doneWithTrans){

**switch** (ui.getTransCode()) {

**case** "IN":

log.displayThis("IN " + ui.getCode() + "," + ui.getId() + ","

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

+ ui.getPopulation() + "," + ui.getLifeExpectancy());

dt.insert1Country(ui.getCode(), ui.getId(), ui.getName(),

ui.getContinent(), ui.getArea(), ui.getPopulation(),

ui.getLifeExpectancy(), log);

ni.insertIntoNameIndex(ui.getName(), ui.getId());

log.displayThis(" OK, country inserted " +

"(in data storage & name index)");

log.displayThis(String.*format*(

" >> %d %s visited in the name index",

ni.getVisited(),

(ni.getVisited() == 1) ? "node": "nodes"));

**break**;

**case** "DN":

log.displayThis("DN " + ui.getName());

ni.deleteByName(dt, ui.getName(), log);

**break**;

**case** "DI":

log.displayThis("DI " + ui.getId());

dt.deleteById(ui.getId(), log);

**break**;

**case** "SN":

log.displayThis("SN " + ui.getName());

ni.selectByName(dt, ui.getName(), log);

**break**;

**case** "SI":

log.displayThis("SI " + ui.getOrigId());

dt.selectById(ui.getId(), (**short**) 0, log, **false**);

**break**;

**case** "AN":

log.displayThis("AN");

ni.selectAllByName(dt, log);

**break**;

**case** "AI":

log.displayThis("AI");

dt.selectAllById(log);

**break**;

**default**:

log.displayThis(ui.getTransCode() + "\n"

+ " ERROR, invalid command");

**break**;

}

ui.grabCommand();

}

N = ui.getTransactions();

ui.finishUp(log);

dt.finishUp();

ni.finishUp();

log.displayThis("CODE STATUS > UserApp finished - " + N

+ " transactions processed");

log.finishUp();

}

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

\* Load NameIdex from Backup.txt

\* **@param** ni

\* **@throws** IOException

\*/

**private** **static** **void** **load**(NameIndex ni) **throws** IOException {

Scanner input = **new** Scanner(**new** File("Backup.txt"));

String[] line = input.nextLine().split("'");

**short** n = Short.*parseShort*(line[0]);

**short** nextEmpty = Short.*parseShort*(line[1]);

**short** rootPtr = Short.*parseShort*(line[2]);

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

line = input.nextLine().split("'");

ni.load(Short.*parseShort*(line[0]), line[1],

Short.*parseShort*(line[2]), Short.*parseShort*(line[3]));

}

ni.loadHeader(n, nextEmpty, rootPtr);

input.close();

}

}

# **package** edu.wmich.cs3310.a2;

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

**import** java.text.DecimalFormat;

**import** java.util.Scanner;

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

\* Countries Of The World App 1.0

\* **@author** Caleb Viola

\*/

**public** **class** PrettyPrintUtility {

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

\* Displays the backup and countrydata files to the log file

\* **@throws** IOException

\*/

**public** **static** **void** **main**() **throws** IOException{

Scanner input = **new** Scanner(**new** File("CountryData.txt"));

PrintWriter log = **new** PrintWriter(**new** FileOutputStream(**new** File(

"Log.txt"), **true**));

String temp;

String[] line = input.nextLine().split("'");

**int** maxId = Integer.*parseInt*(line[1]);

log.println("CODE STATUS > PrettyPrintUtility started");

log.printf("N: %d | MaxId: %d\n", Integer.*parseInt*(line[0]), maxId);

log.println("DATA STORAGE");

log.println("LOC/ CDE ID- NAME-------------- CONTINENT---- ------AREA ---POPULATION LIFE");

**for** (**short** x = 1; x <= maxId; x++) {

temp = input.nextLine();

**if** (Character.*isLetter*(temp.charAt(0))) {

line = temp.split("'");

log.println(String.*format*(

"%03d/ %s %s %-18s %-13s %,10d %,13d %4.1f", x,

line[0], line[1], line[2], line[3],

Integer.*parseInt*(line[4]), Integer.*parseInt*(line[5]),

Float.*parseFloat*(line[6])));

} **else**

log.println(String.*format*("%03d/ %03d ...", x, x));

}

input.close();

input = **new** Scanner(**new** File("Backup.txt"));

line = input.nextLine().split("'");

**int** N = Integer.*parseInt*(line[0]);

log.println("");

log.println("NAME INDEX");

log.printf("N: %s | NextEmpty: %s | RootPtr: %s\n", N, line[1], line[2]);

log.println("LOC/ Lch NAME-------------- PTR Rch");

**for** (**short** x = 0; x < N; x++) {

line = input.nextLine().split("'");

log.println(String.*format*("%03d/ %03d %-18s %03d %03d", x,

Short.*parseShort*(line[0]), line[1],

Short.*parseShort*(line[2]), Short.*parseShort*(line[3])));

}

log.println("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++");

log.println("CODE STATUS > PrettyPrintUtility finished");

input.close();

log.close();

}

}

**import** java.io.File;

**import** java.io.IOException;

**import** java.util.Scanner;

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

\* Read RawData files and organize contents for retrieval

\* Countries Of The World App 1.0

\* **@author** Caleb Viola

\*/

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

**private** Scanner input;

**private** String name;

**private** String continent;

**private** String code;

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

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

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

**private** **int** transactions;

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

**private** Log log;

**boolean** doneWithInput;

**boolean** countryGrabbed;

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

\* Contructor

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

\* **@param** Log TheLog object

\* **@throws** IOException

\*/

**public** **RawData**(String fileNameSufix, Log log) **throws** IOException{

log.displayThis(String.*format*(

"FILE STATUS > RawData FILE opened (A1RawData%s.csv)",

fileNameSufix));

input = **new** Scanner(**new** File("A2RawData"+fileNameSufix+".csv"));

}

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

\* Obtain country from a line from the RawData file

\*/

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

doneWithInput = **false**;

countryGrabbed = **false**;

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

String temp = input.nextLine();

**if** (temp.length() > 2 && temp.substring(0,6).equals("INSERT")){

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

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

code = fields[0];

id = Short.*parseShort*(fields[1]);

name = fields[2];

continent = fields[3];

area = Math.*round*(Float.*parseFloat*(fields[5]));

population = Long.*parseLong*(fields[7]);

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

transactions++;

countryGrabbed = **true**;

}**else**

countryGrabbed = **false**;

}**else**

doneWithInput = **true**;

}

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

\* Getter for code

\* **@return**

\*/

**public** String **getCode**(){

**return** code;

}

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

\* Getter for id

\* **@return**

\*/

**public** **short** **getId**(){

**return** id;

}

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

\* 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 transactions

\* **@return**

\*/

**public** **int** **getTransactions**(){

**return** transactions;

}

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

\* Close file and create log entry

\* **@param** Log TheLog object

\* **@throws** IOException

\*/

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

input.close();

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

transactions = 0;

}

}

**package** edu.wmich.cs3310.a2;

**import** java.io.File;

**import** java.io.IOException;

**import** java.util.Scanner;

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

\* Read TransData files prepare for UserApp

\* Countries Of The World App 1.0

\* **@author** Caleb Viola

\*/

# **public** **class** UI {

**private** Scanner input;

**private** String name;

**private** String continent;

**private** String code;

**private** String transCode;

**private** String origId;

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

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

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

**private** **int** transactions;

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

**boolean** doneWithTrans = **false**;

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

\* Constructor to open TransData and create log entry

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

\* **@param** log

\* **@throws** IOException

\*/

**public** **UI**(**short** fileNameSufix, Log log) **throws** IOException{

log.displayThis(String.*format*(

"FILE STATUS > TransData FILE opened (A1TransData%s.txt)",

fileNameSufix));

input = **new** Scanner(**new** File("A2TransData"+fileNameSufix+".txt"));

}

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

\* Obtain command from a line in the TransData file

\* **@throws** IOException

\*/

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

doneWithTrans = **false**;

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

String temp = input.nextLine();

transCode = temp.substring(0,2);

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

temp = temp.substring(33, temp.length()-2)

.replace("'","");

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

code = fields[0];

id = Short.*parseShort*(fields[1]);

name = fields[2];

continent = fields[3];

area = Math.*round*(Float.*parseFloat*(fields[5]));

population = Long.*parseLong*(fields[7]);

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

}

**else** **if** (transCode.equals("SN") || transCode.equals("DN")){

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

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

**else**

name = "EMPTY";

}**else** **if** (transCode.endsWith("SI") || transCode.endsWith("DI"))

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

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

**try** {

id = Short.*parseShort*(origId.replaceFirst("^0+(?!$)", ""));

}**catch**(Exception e){

id = 0;

}

}

**else**

id = 0;

transactions++;

}

**else**

doneWithTrans = **true**;

}

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

\* Getter for acquired command

\* **@return**

\*/

**public** String **getTransCode**(){

**return** transCode;

}

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

\* Getter for code

\* **@return**

\*/

**public** String **getCode**(){

**return** code;

}

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

\* Getter for id

\* **@return**

\*/

**public** **short** **getId**(){

**return** id;

}

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

\* Getter for origId

\* **@return**

\*/

**public** String **getOrigId**(){

**return** origId;

}

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

\* 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 number of transactions

\* **@return**

\*/

**public** **int** **getTransactions**() {

**return** transactions;

}

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

\* Close file and create log entry

\* **@param** ppu TheLog object

\* **@throws** IOException

\*/

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

input.close();

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

}

}

**package** edu.wmich.cs3310.a2;

**import** java.io.FileNotFoundException;

**import** java.io.IOException;

**import** java.io.RandomAccessFile;

**import** java.util.ArrayList;

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

\* Countries Of The World App 1.0

\* **@author** Caleb Viola

\*/

# **public** **class** DataTable {

**private** **short** n;

**private** **short** maxId;

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

**private** RandomAccessFile file;

**private** DataTableRecord dtr;

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

\* Constructor

\* **@param** fileName

\* **@param** log

\* **@throws** IOException

\*/

**public** **DataTable**(String fileName, Log log) **throws** IOException{

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

file.seek(0);

**try** {

String[] line = file.readLine().split("'");

n = Short.*parseShort*(line[0]);

maxId = Short.*parseShort*(line[1]);

} **catch** (Exception e) {

n = 0;

maxId = 1;

file.writeBytes(String.*format*("%03d'%03d\n", n, maxId));

}

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

dtr = **new** DataTableRecord();

}

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

\* IN command

\* **@param** code

\* **@param** id

\* **@param** name

\* **@param** continent

\* **@param** area

\* **@param** population

\* **@param** lifeExpectancy

\* **@param** userApp

\* **@param** log

\* **@throws** IOException

\*/

**public** **void** **insert1Country**(String code, **short** id, String name,

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

Log log) **throws** IOException {

dtr.byteOffset(id);

status = dtr.locateWithStatus(file);

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

**if** (id > maxId){

/\* fill space in between with pre-formatted string

\* for better management and readability

\* \*/

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

dtr.byteOffset(maxId+1);

status = dtr.locateWithStatus(file);

**for** (**int** x = maxId+1; x < id; x++)

dtr.fill1record(file);

}

maxId = id;

}

dtr.write1Country(file, code, id, name, continent, area,

population, lifeExpectancy);

n++;

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

log.displayThis(" SORRY, another country has that id");

**else**

log.displayThis(" SORRY, invalid id");

}

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

\* DI command

\* **@param** id

\* **@param** log

\* **@throws** IOException

\*/

**public** **void** **deleteById**(**short** id, Log log) **throws** IOException{

log.displayThis(" [SORRY, Delete By Id module not yet working]");

}

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

\* Execution for SI command which locates an

\* element in a binary file using id

\* **@param** id

\* **@param** visited

\* **@param** log

\* **@param** nameIndex

\* **@throws** IOException

\*/

**public** **void** **selectById**(**short** id, **short** visited, Log log, **boolean** nameIndex) **throws** IOException{

dtr.byteOffset(id);

status = dtr.locateWithStatus(file);

**if**(id == 0)

log.displayThis(" SORRY, no country with that id");

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

dtr.read1Country(file);

log.displayThis(" "

+ log.country(dtr.getCode(), dtr.getId(), dtr.getName(),

dtr.getContinent(), dtr.getArea(),

dtr.getPopulation(), dtr.getLifeExpectancy()));

**if**(nameIndex)

log.displayThis(String.*format*(" >> %d %s visited",

visited, (visited == 1) ? "node" : "nodes"));

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

log.displayThis(" SORRY, invalid id");

**else**

log.displayThis(" SORRY, no country with that id");

}

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

\* AI command which displays table in id order

\* **@param** log

\* **@throws** IOException

\*/

**public** **void** **selectAllById**(Log log) **throws** IOException{

log.displayThis(" "+log.header());

**short** i = 1;

dtr.byteOffset(i);

status = dtr.locateWithStatus(file);

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

dtr.read1Country(file);

**if**(dtr.getId() != 0)

log.displayThis(" "

+ log.country(dtr.getCode(), dtr.getId(),

dtr.getName(), dtr.getContinent(),

dtr.getArea(), dtr.getPopulation(),

dtr.getLifeExpectancy()));

dtr.byteOffset(++i);

status = dtr.locateWithStatus(file);

}

log.displayThis(" "+log.footer());

}

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

\* Dumps dataTable into Backup.txt

\* **@throws** IOException

\*/

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

file.seek(0);

file.writeBytes(String.*format*("%03d'%03d\n", n, maxId));

file.close();

}

}

**package** edu.wmich.cs3310.a2;

**import** java.io.IOException;

**import** java.io.RandomAccessFile;

**import** java.text.DecimalFormat;

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

\* Manages file handling

\* Countries Of The World App 1.0

\* **@author** Caleb Viola

\*/

# **public** **class** DataTableRecord {

**private** String code;

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

**private** String name;

**private** String continent;

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

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

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

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

**private** **int** sizeOfHeaderRec = 3 /\*n\*/ + 1 /\*single quote\*/ + 3 /\*maxId\*/ + 1 /\*line feed\*/;

**private** **int** sizeOfDataRec = 3 /\*code\*/ + 1 /\*single quote\*/ + 3 /\*id\*/ + 1 /\*single quote\*/ +

15 /\*name\*/ + 1 /\*single quote\*/ + 13 /\*continent\*/ + 1 /\*single quote\*/ + 8 /\*area\*/ + 1 /\*single quote\*/ + 10 /\*population\*/ + 1 /\*single quote\*/ + 4 /\*lifeExpectancy\*/ + 1 /\*line feed\*/;

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

\* For reading a country from record

\* **@param** file

\* **@throws** IOException

\*/

**public** **void** **read1Country**(RandomAccessFile file) **throws** IOException{

String[] line = file.readLine().split("'");

code = line[0];

id = Short.*parseShort*(line[1]);

name = line[2].trim();

continent = line[3].trim();

area = Integer.*parseInt*(line[4]);

population = Long.*parseLong*(line[5]);

lifeExpectancy = Float.*parseFloat*(line[6]);

}

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

\* 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** **write1Country**(RandomAccessFile file, String code, **short** id,

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

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

file.writeBytes(String.*format*(

"%s'%03d'%-15.15s'%-13.13s'%08d'%010d'%4.1f\n", code, id, name,

continent, area, population, lifeExpectancy));

}

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

\* Fills empty spaces with 0s

\* **@param** file

\* **@throws** IOException

\*/

**public** **void** **fill1record**(RandomAccessFile file) **throws** IOException{

file.writeBytes(String.*format*(

"%-3.3s'%03d'%-15.15s'%-13.13s'%08d'%010d'%04.1f\n", " ", 0,

" ", " ", 0, 0, 0.0));

}

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

\* Discover nature of file at given offset and return status

\* **@param** file

\* **@return**

\*/

**public** **boolean**[] **locateWithStatus**(RandomAccessFile file){

**try** {

file.seek(byteOffset);

**try** {

**char** test = (**char**)file.readByte();

file.seek(byteOffset);

**if** (Character.*isLetter*(test))

**return** **new** **boolean**[] { **true**, **false** }; // Occupied spot

**return** **new** **boolean**[] { **false**, **true** }; // Empty spot

} **catch** (IOException e) {

**return** **new** **boolean**[] { **true**, **true** };// New spot

}

} **catch** (IOException e) {

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

}

}

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

\* Calculate offset

\* **@param** rrn

\*/

**public** **void** **byteOffset**(**int** rrn){

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

}

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

\* Getter for code

\* **@return**

\*/

**public** String **getCode**(){

**return** code;

}

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

\* Getter for id

\* **@return**

\*/

**public** **short** **getId**(){

**return** id;

}

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

\* 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.a2;

**import** java.io.IOException;

**import** java.util.ArrayList;

**import** java.util.jar.Attributes.Name;

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

\* Countries Of The World App 1.0

\* **@author** Caleb Viola

\*/

# **public** **class** NameIndex {

**private** **short** n, nextEmpty, rootPtr, visited;

**private** **final** **short** MAX\_N\_HOME\_LOC = 40;

**private** BSTNode[] bstNodes;

**public** **NameIndex**(){

n = 0;

nextEmpty = 0;

rootPtr = -1;

bstNodes = **new** BSTNode[MAX\_N\_HOME\_LOC];

}

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

\* insertIntoNameIndex

\* **@param** keyValue

\* **@param** drp

\*/

**public** **void** **insertIntoNameIndex** (String keyValue, **short** drp){

bstNodes[nextEmpty] = **new** BSTNode();

bstNodes[nextEmpty].name = keyValue;

bstNodes[nextEmpty].drp = drp;

bstNodes[nextEmpty].leftChPtr = -1;

bstNodes[nextEmpty].rightChPtr = -1;

visited = 0;

**if** (rootPtr == -1)

rootPtr = 0;

**else** {

**int** i = rootPtr;

**int** parentI = rootPtr;

String side = "na";

**while** (i != -1) {

parentI = i;

**if** (bstNodes[nextEmpty].name.compareTo(bstNodes[i].name) < 0){

i = bstNodes[i].leftChPtr;

side = "left";

}

**else** **if** (bstNodes[nextEmpty].name.compareTo(bstNodes[i].name) > 0){

i = bstNodes[i].rightChPtr;

side = "right";

}

**else**{

bstNodes[nextEmpty] = **null**;

nextEmpty--;

i = -1;

}

visited++;

}

**if** (side.equals("left"))

bstNodes[parentI].leftChPtr = nextEmpty;

**else** **if** (side.equals("right"))

bstNodes[parentI].rightChPtr = nextEmpty;

}

n++;

nextEmpty++;

}

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

\* Loads directly to index

\* **@param** keyValue

\* **@param** drp

\*/

**public** **void** **load** (**short** Lch, String keyValue, **short** drp, **short** Rch){

bstNodes[nextEmpty] = **new** BSTNode();

bstNodes[nextEmpty].name = keyValue;

bstNodes[nextEmpty].drp = drp;

bstNodes[nextEmpty].leftChPtr = Lch;

bstNodes[nextEmpty].rightChPtr = Rch;

nextEmpty++;

}

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

\* Load record header data

\* **@param** n

\* **@param** nextEmpty

\* **@param** rootPtr

\*/

**public** **void** **loadHeader**(**short** n, **short** nextEmpty, **short** rootPtr){

**this**.n = n;

**this**.nextEmpty = nextEmpty;

**this**.rootPtr = rootPtr;

}

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

\* SN command with binary search

\* **@param** dt

\* **@param** name

\* **@param** log

\* **@throws** IOException

\*/

**public** **void** **selectByName** (DataTable dt, String name, Log log) **throws** IOException{

**short**[] i = searchForName(name);

**if** (i[0] != -1)

dt.selectById(bstNodes[i[0]].drp, i[1], log, **true**);

**else**

log.displayThis(String.*format*(

" ERROR, invalid country name\n >> %d %s visited",

visited, (visited == 1) ? "node" : "nodes"));

}

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

\* Searches for target in the BST

\* **@param** target

\* **@return**

\*/

**private** **short**[] **searchForName**(String target){

**short** i = rootPtr;

visited = (**short**)(rootPtr+1);

**while** (i != -1 && !target.equals(bstNodes[i].name)) {

**if** (target.compareTo(bstNodes[i].name) < 0)

i = bstNodes[i].leftChPtr;

**else**

i = bstNodes[i].rightChPtr;

visited++;

}

**short**[] result = {i, visited};

**return** result;

}

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

\* DN command

\* **@param** dr

\* **@param** name

\* **@param** log

\* **@throws** IOException

\*/

**public** **void** **deleteByName** (DataTable dt, String name, Log log) **throws** IOException{

log.displayThis(" [SORRY, Delete By Name module not yet working]");

}

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

\* AN command

\* **@param** dt

\* **@param** log

\* **@throws** IOException

\*/

**public** **void** **selectAllByName**(DataTable dt, Log log) **throws** IOException {

log.displayThis(" "+log.header());

LNR(rootPtr, dt, log);

log.displayThis(" "+log.footer());

}

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

\* Inorder traversal algorithm using recursion

\* **@param** i Search index

\* **@param** tl TheLog object

\* **@throws** IOException

\*/

**private** **void** **LNR**(**short** i, DataTable dt, Log log) **throws** IOException{

**if**(bstNodes[i].leftChPtr != -1){

LNR(bstNodes[i].leftChPtr, dt, log);

dt.selectById(bstNodes[i].drp, (**short**)0, log, **false**);

}**else**

dt.selectById(bstNodes[i].drp, (**short**)0, log, **false**);

**if**(bstNodes[i].rightChPtr != -1){

LNR(bstNodes[i].rightChPtr, dt, log);

}

}

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

\* Getter for visited

\* **@return**

\*/

**public** **short** **getVisited**(){

**return** visited;

}

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

\* Dumps nameIndex to Backup.txt

\* **@throws** IOException

\*/

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

Backup backup = **new** Backup();

backup.displayThis(String.*format*("%d'%d'%d", n, nextEmpty, rootPtr));

**for** (**short** x = 0; x < n; x++)

backup.displayThis(backup.nameFormat(bstNodes[x].leftChPtr,

bstNodes[x].name, bstNodes[x].drp, bstNodes[x].rightChPtr));

backup.finishUp();

}

}

**package** edu.wmich.cs3310.a2;

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

\* Countries Of The World App 1.0

\* DataTable node

\* **@author** Caleb Viola

\*/

# **public** **class** BSTNode {

String name;

**short** drp;

**short** leftChPtr, rightChPtr;

}

**package** edu.wmich.cs3310.a2;

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

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

\* Manages Log.txt

\* Countries Of The World App 1.0

\* **@author** Caleb Viola

\*/

# **public** **class** Log {

**private** PrintWriter log;

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

\* Opens Log.txt in append or truncate mode

\* **@param** append

\* **@throws** IOException

\*/

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

**if** (!append)

log = **new** PrintWriter("Log.txt");

**else**

log = **new** PrintWriter(**new** FileOutputStream(**new** File("Log.txt"),

append));

log.println("FILE STATUS > Log FILE opened");

}

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

\* Formatted string for printing country

\* **@param** code

\* **@param** id

\* **@param** name

\* **@param** continent

\* **@param** area

\* **@param** population

\* **@param** lifeExpectancy

\* **@return**

\*/

**public** String **country**(String code, **short** id, String name, String continent,

**int** area, **long** population, **float** lifeExpectancy) {

**return** String.*format*("%s %03d %-18.18s %-13.13s %,10d %,13d %4.1f",

code, id, name, continent, area, population, lifeExpectancy);

}

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

\* Formatted string for printing empty location

\* **@param** id

\* **@return**

\*/

**public** String **empty**(**int** id){

**return** String.*format*(" %03d ...", id);

}

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

\* Formatted string for header in AI and AN

\* **@return** SA header string

\*/

**public** String **header**(){

**return** "CDE ID- NAME-------------- CONTINENT---- ------AREA ---POPULATION LIFE";

}

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

\* Formatted screen for footer in AI and AN

\* **@return** footer

\*/

**public** String **footer**(){

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

}

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

\* Prints line to file

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

\* **@throws** IOException

\*/

**public** <**T**> **void** **displayThis**(**T** message) **throws** IOException{

log.println(message);

}

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

\* Creates log entry and closes file

\* **@throws** IOException

\*/

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

log.println("FILE STATUS > Log FILE closed");

log.close();

}

}

**package** edu.wmich.cs3310.a2;

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

**import** java.text.DecimalFormat;

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

\* Manages Backup.txt

\* Countries Of The World App 1.0

\* **@author** Caleb Viola

\*/

# **public** **class** Backup {

**private** PrintWriter backup;

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

\* Constructor

\* **@throws** IOException

\*/

**public** **Backup**() **throws** IOException{

backup = **new** PrintWriter("Backup.txt");

}

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

\* String format for nameIndex print on backup file

\* **@param** Lch

\* **@param** name

\* **@param** ptr

\* **@param** Rch

\* **@return**

\*/

**public** String **nameFormat**(**short** Lch, String name, **short** ptr, **short** Rch){

**return** String.*format*("%d'%s'%d'%d", Lch, name, ptr, Rch);

}

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

\* Prints line to file

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

\* **@throws** IOException

\*/

**public** <**T**> **void** **displayThis**(**T** message) **throws** IOException{

backup.println(message);

}

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

\* Closes backup file

\* **@throws** IOException

\*/

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

backup.close();

}

}