/\*

\* Class: StormChaser.java

\* Purpose: Main program to read hurricane data file, create Storm objects,

\* and keep those that are category 3 or higher.

\*/

import java.io.\*;

import java.util.Scanner;

public class StormChaser {

public static void main(String[] args)

{

// Constants

final int MAX\_STORMS = 300;

// array of Storms

Storm CurrentStorm; // storm returned by GetStorm

int nStorms = 0; // number in array List

int totalStorms = 0; // total number of storms in the input file

Storm[] List = new Storm[MAX\_STORMS];

Scanner fileInput;

// Openning hurricane data file

try{

System.out.println("Openning hurricane data file...");

fileInput = new Scanner(new File("hurricanedata1950to2015.txt"));

}

catch(FileNotFoundException e){

System.err.println("FileNotFoundException: " + e.getMessage());

return;

}

System.out.println( "File opened successfully...");

System.out.println( "Reading file..." );

// Read Storm data from file until EOF

while(fileInput.hasNextLine())

{

CurrentStorm = GetStorm(fileInput);

++totalStorms;

// If Storm i category 3 or higher, add to the array

if(CurrentStorm.getCategory()>= 3)

{

List[nStorms] =CurrentStorm;

nStorms++;

}

}

System.out.println( "Number of storms: " + totalStorms );

System.out.println( "Hurricanes with category 3 and above: " + nStorms );

DisplayStorms( "First Twenty Storms", List, 20 );

Sort( List, nStorms );

DisplayStorms( "Top Twenty Storms", List, 20 );

fileInput.close();

}

public static Storm GetStorm( Scanner fileInput )

{

// Create constants as array indexes for data elements

final int STORMID = 0;

final int NAME = 1;

final int MAXRECORDS = 2;

final int BEGINDATE = 0;

final int WIND = 6;

final int PRESSURE = 7;

// Declare variables

int wind = 0, pressure = 0, maxRecords = 0;

int beginDate = 0, duration = 0;

String name;

String stormID;

String header;

String data;

Storm NewStorm;

// Read the Storm header information

header = fileInput.nextLine();

// Tokenize the header

String[] headerElements = header.split(",");

stormID = headerElements[STORMID].trim();

name = headerElements[NAME].trim();

maxRecords = Integer.parseInt(headerElements[MAXRECORDS].trim());

// Read first row of Storm data

data = fileInput.nextLine();

// Tokenize the Storm data

String[] dataElements = data.split(",");

// Extract the data elements

beginDate = Integer.parseInt(dataElements[BEGINDATE].trim());

wind = Integer.parseInt(dataElements[WIND].trim());

pressure = Integer.parseInt(dataElements[PRESSURE].trim());

duration = 6;

// Create Storm object

NewStorm = new Storm(stormID, beginDate, duration, name, wind, pressure);

for( int i = 1; i < maxRecords; i++ )

{

// Read next row of Storm data

data = fileInput.nextLine();

// Tokenize the Storm data

dataElements = data.split(",");

// Extract the data elements

wind = Integer.parseInt(dataElements[WIND].trim());

pressure = Integer.parseInt(dataElements[PRESSURE].trim());

duration += 6;

// Update Storm object

NewStorm.setPressure(pressure);

NewStorm.setWind(wind);

NewStorm.setDuration(duration);

}

// Return the new storm object

return NewStorm;

}

public static void DisplayStorms( String title, Storm[] List, int NStorms )

{

// display NStorms storms

// print some title and column headings

System.out.println(title);

System.out.println(

" Begin Duration Maximum Minimum ");

System.out.println(

"Storm ID Date (hours) Name Category Winds (mph) Press. (mb)");

System.out.println(

"-------------------------------------------------------------------------");

for( int k = 0; k < NStorms; k++ )

{

// Print out one Storm

System.out.print(List[k].toString());

}

System.out.println ("\n");

}

public static void Sort( Storm StormList[], int n )

{

// bubble sort the list of Storms

int pass = 0, k, switches;

Storm temp;

switches = 1;

while( switches != 0 )

{

switches = 0;

pass++;

for( k = 0; k < n - pass; k++ )

{

if( StormList[k].getCategory() < StormList[k+1].getCategory() )

{

temp = StormList[k];

StormList[k] = StormList[k+1];

StormList[k+1] = temp;

switches = 1;

}

}

}

}

}

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

/\*\*

\*

\* @author Cameron Willliams

\*/

/\*

\* Class: Storm.java

\* Purpose: Store hurricane data for individual storm objects

\*/

public class Storm {

private final double KnotsToMPH = 1.15;

private String stormID;

private int beginDate;

private int duration;

private String name;

private int category;

private int wind;

private int pressure;

public Storm( String stormid, int bdate, int dur, String sname, int w, int p )

{

stormID = stormid;

beginDate = bdate;

duration = dur;

name = sname;

wind = (int)(w \* KnotsToMPH);

pressure = p;

SaffirSimpson();

}

public void setDuration( int d )

{

duration = d;

}

public void setWind( int w )

{

w = (int)(w \* KnotsToMPH);

if(w > wind)

wind = w;

SaffirSimpson();

}

public void setPressure( int p )

{

if(pressure <= 0)

pressure = p;

if(p > 0 && p < pressure)

pressure = p;

SaffirSimpson();

}

public void SaffirSimpson()

{

// Compute storm category, using the Saffir-Simpson scale

if(pressure <= 920 && wind >= 156)

{

category = 5; // Category 5

}

if(pressure > 920 && wind < 156)

{

category = 4; // Category 4

}

if(pressure > 945 && wind < 113)

{

category = 3; // Category 3

}

if(pressure > 965 && wind < 96)

{

category = 2; // Category 2

}

if(pressure > 980 && wind < 83)

{

category = 1; // Category 1

}

if(wind < 64)

{

category = -1; // Tropical Storm

}

if(wind < 34)

{

category = -2; // Tropical Depression

}

if(pressure == 0)

{

category = 0; // Missing pressure

}

}

public int getCategory()

{

return category;

}

public String toString()

{

return String.format("%s %9d %7d %-10s %2d %6d %8d\n",

stormID, beginDate, duration, name, category, wind, pressure);

}

}

run:

Openning hurricane data file...

File opened successfully...

Reading file...

Number of storms: 998

Hurricanes with category 3 and above: 249

First Twenty Storms

Begin Duration Maximum Minimum

Storm ID Date (hours) Name Category Winds (mph) Press. (mb)

-------------------------------------------------------------------------

AL011950 19500812 306 ABLE 4 126 953

AL021950 19500818 360 BAKER 3 103 979

AL031950 19500821 366 CHARLIE 3 109 974

AL041950 19500830 444 DOG 4 143 953

AL051950 19500901 234 EASY 4 120 960

AL061950 19500908 228 FOX 4 138 946

AL071950 19500927 252 GEORGE 3 109 960

AL091950 19501008 96 ITEM 3 103 980

AL111950 19501013 186 KING 4 132 955

AL041951 19510812 312 CHARLIE 4 132 968

AL061951 19510901 330 EASY 4 149 937

AL091951 19510929 318 HOW 3 97 963

AL021952 19520818 396 ABLE 3 97 983

AL041952 19520831 240 BAKER 3 109 969

AL061952 19520924 174 CHARLIE 4 120 958

AL091952 19521006 126 EASY 3 103 968

AL101952 19521020 216 FOX 4 143 934

AL041953 19530828 312 CAROL 4 161 929

AL071953 19530915 168 EDNA 4 114 962

AL091953 19530923 114 FLORENCE 4 114 968

Top Twenty Storms

Begin Duration Maximum Minimum

Storm ID Date (hours) Name Category Winds (mph) Press. (mb)

-------------------------------------------------------------------------

AL101955 19550921 240 JANET 5 172 914

AL091961 19611027 120 HATTIE 5 161 920

AL091969 19690814 222 CAMILLE 5 172 900

AL041980 19800731 276 ALLEN 5 189 899

AL081988 19880908 294 GILBERT 5 184 888

AL111989 19890910 384 HUGO 5 161 918

AL131998 19981022 468 MITCH 5 178 905

AL132003 20030906 348 ISABEL 5 166 915

AL092004 20040902 564 IVAN 5 166 910

AL122005 20050823 204 KATRINA 5 172 902

AL182005 20050918 216 RITA 5 178 895

AL252005 20051015 288 WILMA 5 184 882

AL042007 20070813 252 DEAN 5 172 905

AL011950 19500812 306 ABLE 4 126 953

AL041950 19500830 444 DOG 4 143 953

AL051950 19500901 234 EASY 4 120 960

AL061950 19500908 228 FOX 4 138 946

AL111950 19501013 186 KING 4 132 955

AL041951 19510812 312 CHARLIE 4 132 968

AL061951 19510901 330 EASY 4 149 937

BUILD SUCCESSFUL (total time: 0 seconds)