Horses and Burros

Purpose: To work with serialized objects and CSV files and incorporate many elements previously covered in class

Learning Objectives:

- To get exposure to CSV files that contain data about objects
- To use buffered readers to read a CSV file
- To get a chance to practice with random number generation
- To create and use custom exceptions
- To practice using String methods to parse a line of data
- To serialize and deserialize instances of a Class
- To practice working with enums

Background:

This application will upload statistics for USA Herd Management data. The *Wild and Free-Roaming Horses and Burros Act of 1971* (WFRHBA), is an Act of Congress, signed into law by President Richard M. Nixon on December 18, 1971. The act covered the management, protection and study of "unbranded and unclaimed horses and burros on public lands in the United States." (Wikipedia) It also established Herd Areas (HAs) – geographic areas where wild horses and/or burros were found at the passage of the Act – and Herd Management Areas (HMAs) within Herd Areas where Land Use Plans will manage the populations of wild horses and/or burros.

This lab starts with reading the most recent data in a CSV or comma-separated file. CSV files are common formats to transmit data. They are easily produced by Microsoft Excel and a wide variety of applications.

Instructions:

1. Acquaint yourself with the starting materials for the lab:
   - **Statistic.java:** a generic superclass which can have many subclasses based on different record types;
   - **StateStatistic.java:** a class which describes the summary horse/burro data for each state covered by the Act;
   - **State. java:** an enum which contains a value for the states covered by the Act;
   - **DataSet. java:** a class which contains a collection of statistics;

   Note the Javadoc for these classes.

   - **herdManagement.csv**: the comma separated file.

   The comma separated file looks like this:

```
Herd Management Records by State,,,,,,
,,,,,,
State,HerdAreaAcresBLM,HerdAreaAcresOther,HerdManagementAreaAcresBLM,HerdManagementAreaAcresOther,NumHorses,NumBurros
AZ,2019932,1617998,1756086,1327777,390,1967
CA,4810248,1813228,1946590,471855,4057,895
CO,658119,76572,366098,38656,772,0
ID,428421,49235,377907,40287,913,0
MT,104361,119242,28282,8865,195,0
NV,19076183,3073205,13580401,1668864,16642,819
NM,88653,37874,24505,4107,114,0
OR,3565790,773754,2703409,259726,2508,15
UT,3150220,676855,2174850,310747,2495,142
WY,6389094,2786158,3638330,1137121,5016,0
```

   Note that line 1 of the file contains a description of the data, line 2 of the file is a blank row, and line 3 of the file contains the column headings. The actual data begins on line 4 with Arizona data showing 390 wild horses and1967 wild burros.

2. Your main method in the Driver class should:

   - Create an instance of the DataSet class;
   - Invoke a method named `loadStatistics()` which accepts as parameters:
     i. The DataSet you wish to populate
     ii. The name of the file containing the data
     iii. The number of "header rows" that the data file contains (i.e., descriptive rows *not* containing data)
   - Invoke a method named `displayStatistics()` which will display one row for each StateStatistic instance loaded into the dataset.

   Build a serialize() and deserialize method. ***THINK CAREFULLY about which class should contain these methods.***

   - Select a random StateStatistic to serialize.
   - Invoke `serialize()` to serialize a StateStatistic instance to an object file.
   - Create a new StateStatistic and assign it with the deserialized version of the statistic in the previous step by invoking the method `deserialize().`
   - Output the number of wild horses and burros for the deserialized statistic, such as

     ```
     There are 819 burros and 16642 horses in NV.
     ```

3. The method `loadStatistics()` should operate as follows:

   - Read the data file using a buffered reader.
   - Read and disregard any header rows.
   - Read each line of data and split the line into an array of Stings. See http://pages.cs.wisc.edu/~hasti/cs302/examples/Parsing/parseString.html for some good examples.
   - Utilize the values of this array to instantiate a StateStatistic object.
     (How will you convert Strings to enum values and long datatypes?)
   - Add the statistic to the DataSet instance you created above.

4. Finally, create a custom exception StatisticDataNotFoundException. This exception should be thrown whenever you cannot find the filename that contains the data. This exception should:
   - Hold the value of the filename
   - Hold the timestamp of when the file was not available. (You can use `LocalDate.now()` for this.)

5. Include both a UML class diagram and a UML sequence diagram that shows all interactions between classes.