Fruit Basket App

Overview

You will be building a console application (CLI) that allows a user to process a CSV-file and produce a summary report from it.

This exercise should take 2 - 4 hours to complete.

Your code should meet the functional requirements described below, should contain instructions for deployment and running the application, and should be clear and easy to understand.

Your solution may be written in the language of your choosing.

Feel free to use the internet and any other resources to arrive at a reasonable algorithm. However, please do not "cut and paste" a solution. It is more important to have a solution that is easy to understand and run that may not satisfy all of the test cases than to have a poorly organized solution without any tests.
Think about testing strategies.

Functional Requirements

- Produce a text report, executed on and output to the command line.

A user should be able to create the report by providing the path to the CSV-file to the application.
The CSV-file represents a catalog of all the fruit in a fruit bowl.
Each line is a piece of fruit. The data representing a particular fruit include:

- fruit type
- age in days
- ad-hoc characteristic 1
- ad-hoc characteristic 2

leading to the following format:

```
fruit-type,age-in-days,characteristic1,characteristic2
apple,1,red,sweet
orange,2,round,sweet
pineapple,5,prickly,sweet
apple,4,yellow,sweet
```

```
grapefruit,2,yellow,bitter
watermelon,4,green,heavy
orange,2,round,sweet
orange,1,round,sweet
pineapple,6,prickly,sweet
apple,1,green,tart
grapefruit,1,bitter,yellow
watermelon,2,heavy,green
grapefruit,2,bitter,yellow
watermelon,3,heavy,green
orange,1,round,sweet
orange,5,sweet,round
pineapple,2,sweet,prickly
apple,2,red,sweet
orange,6,round,sweet
pineapple,2,sweet,prickly
apple,1,red,sweet
grapefruit,3,yellow,bitter
```

The application should enforce the following rules upon execution:

- If the file does not exist the application should respond with an appropriate exit code, and a message explaining that the file does not exist.
- If the file structure does not match the CSV structure outlined above the application should respond with an
  appropriate exit code, and a message explaining the issue.

If the above conditions are met the application should invoke your algorithm to produce the following summary report:

- Total count of all fruits in the CVS-file
- Total count of distinct fruit **types** in the basket
- The fruit type and age of the oldest fruit in the basket in days
- Count of all fruits grouped by fruit types in descending order
- Count of all fruits grouped by fruit type and all characteristics in descending order

```
Total number of fruit:
22

Total types of fruit:
5

Oldest fruit & age:
orange: 6
pineapple: 6

The number of each type of fruit in descending order:
orange: 6
```

```
apple: 5
pineapple: 4
grapefruit: 4
watermelon: 3

The various characteristics (count, color, shape, etc.) of each fruit by type:
3 apple: red, sweet
5 orange: round, sweet
2 pineapple: prickly, sweet
1 apple: yellow, sweet
2 grapefruit: yellow, bitter
1 watermelon: green, heavy
1 apple: green, tart
2 grapefruit: bitter, yellow
2 watermelon: heavy, green
1 orange: sweet, round
2 pineapple: sweet, prickly
```

Bonus

The following are *not* required but might be nice additions to the exercise:

- Allow for a dynamic amount of characteristic2 to be specified in the CSV:

```
fruit-type,age-in-days,characteristic1,characteristic2,characteristic3,characteristic4
,characteristic5
apple,1,round,sweet,red,year-round
orange,2,round,sweet
pineapple,5,prickly,sweet,yellow
```

- Create a `-help` parameter for your application that provides information on how to use the application
- Create a script that exercises and verifies the functionality of your application
  by executing and verifying that the results are appropriate