Assignment brief

Time 3 hr (+1 hr for upload)

Top Trumps is a popular children's trading card game. Collectable cards typically consist of an image of a character with some information about them, and a series of numerical statistics (speed, strength etc) which can be compared across cards.
A turn based game can be played with these cards where players take a stack of cards each and alternately declare a stat to be compared from the card on top. The best (highest) stat wins and the winning player takes those cards and adds them to the bottom of their stack. Play continues until one player has lost all their cards.

You have been given some code that is currently being developed as part of a superhero themed **Trading Card Game System**. The system will allow analysis of available card data, as well as facilitating the simulation of turned based game using that data.
Not all the requirements have been implemented. It is your task to implement these and raise the coding standards of all the code.

Create a **project solution** (named **<Your Name>_<Student Number>_p3 e.g. MatthewCollins_12345678_P3**). Create a **package** named **p3**.

3 files initial files have been provided.
Add **StartApp.java** and **CardGame.java** to the p3 package in the solution and add the **toptrumpscards.csv** file at the project root.
Ensure your name and student number are placed in the Javadoc comments of all the classes you create.

- The StartApp has been partially completed with a menu and a partially completed readData method.
- CardGame.java is fully complete and **WILL NOT require modification**. It assumes the existence of a **TopTrumpCard** class which you will need to implement for CardGame to fully compile. (Calls to CardGame methods in StartApp have been temporarily commented out/disabled to allow work to continue on other tasks if necessary).

The application will run (start) from the StartApp.java, initially reading in the data from the **toptrumpscards.csv** file and then perform a number of menu driven operations.

Assessment Structure

3 Parts. Weighted 30% each with the remaining 10% for general approach and style.

Part 1: Implement the **TopTrumpCard** Class based on the structure of the data in the csv file, the expectations of the **CardGame** class and the specification details given in this document.

Part 2: Appropriately Unit Test the **TopTrumpCard** Class. Further details later in this document.

Part 3: Edit and further develop **StartApp.java** to read the data from the file into a JCF List of TopTrumpCard objects and implement the missing menu driven functionality.

Part 1 – TopTrumpCard class implementation

Using your knowledge of OOP you should add/update code based on the following:

Analyse the data in the **toptrumpscards.csv** and create a class (**TopTrumpCard.java**) and any other associated files/classes you deem necessary for your design.

Each column in the file corresponds directly with an expected attribute of the object.
The following allowable values/business rules have been agreed.

Attribute	Allowable Values
Name	At least 1 character but no more than 30
Real Name	At least 1 character but no more than 30
Image File Name	At least 5 characters, no more than 30, no space characters and must end with .jpg
Category	Hero or Villain
Speed	Whole number 0-100 inclusive
Strength	Whole number 0-100 inclusive
Agility	Whole number 0-100 inclusive
Intelligence	Whole number 0-100 inclusive
Bio	Any length but cannot be null

Appropriate getters/setters should be provided for all attributes.

An attempt to set a value outside allowable ranges should give an appropriate exception with an appropriate exception message e.g. “Invalid image name” etc.

The implementation of CardGame.java also assumes that the TopTrumpCard class will have the following 2 methods which you will need to implement.

getMaxStatID()	This method should return 0,1,2 or 3 depending on whether Speed, Strength, Agility or Intelligence have the highest value respectively. In the event of multiple values being equally high then any such value can be returned. (developer choice)
getStatScore(statChoice)	Should expect to receive an input of 0, 1 2 or 3 and return the actual value of the corresponding attribute Speed, Strength, Agility or Intelligence

Other methods etc. can be added as the developer deems necessary.

Part 2 – Unit Testing Expectation

Conduct appropriate Unit Testing of the behaviour of the **TopTrumpCard** Class and its methods.

For the purposes of this assessment it is not necessary to test any other classes/methods.

It is recognised that there is some repetition in the requirements of some of the fields, mainly the 4 stat fields.

Therefore in the interest of time saving **it will be acceptable to fully and rigorously test the functionality just one of those stat fields.** For the other fields simple verification of the getter/setter setting values will be enough. (in the real world this should be fully tested but can be skipped in this time limited scenario without penalty)

It is not necessary to test void methods which have no impact on the state of the object. (e.g. printDetails type methods)

If fully autogenerated methods such as equals() are deemed necessary and included in your implementation it will not be expected that unit tests of these are included for the purposes of the assessment.

Part 3 – StartApp and Menu Options

1) The **readData** method has been partially completed. Finish the implementation of this method so that it will work with your **TopTrumpCard** class. (readData method puts card values in a variable called **mainDeck** which will then be used by other methods in the program).

2) Play demo game – Assuming your **TopTrumpCard** class has been implemented as expected the **CardGame** class's **playGame** method can be called from the menu (<u>just uncomment the relevant line of code</u>)

3) Display the **number of Cards** in the Current Deck (repeatable as may change due to other menu options)
   Sample output:
   ```
   Number of cards in current Deck: 18
   ```

4) Display **full details** for all cards in the current Deck.
   Sample output:

```
name:           Spider-Man
realname:       Miles Morales
filename:       miles.jpg
category:       HERO
speed:          82
strength:       77
agility:        90
intelligence:   70
bio:            In his world Peter Parker died and Miles became the
new Spider-Man.

Etc.
```

5) Display **summary details** (name and key info) of the **top 5 strongest Heroes** in the current Deck

Sample output (not necessarily the top values in the actual file):

```
1)
The Incredible Hulk / Bruce Banner (HERO)
sp: 75 | st: 95 | ag: 70 | in: 55

2)
Superman / Clark Kent (HERO)
sp: 85 | st: 90 | ag: 64 | in: 75

3)
Etc. etc.
```

6) Display summary details of **Villains** in the current Deck with an **agility rating of 75 or more**, sorted **alphabetically by name** (same layout as other methods using summary details)

7) Display the card **name and bio** of the card/cards with the **longest** bio in the current Deck (determined by number of characters in the bio – no word count required)

8) Find Swaps. Top Trumps is a trading card game and as such it is possible to have duplicates of cards. Identify and **display summary details of any duplicate cards** in the current Deck.

9) **Remove any duplicate cards** from the current Deck. (The effect of this can be verified with other menu options eg. Option 3 or 4. And can be reversed by reading the file again with Option 1)

10) Sort the deck from **highest to lowest determined by average of the 4 main stats**. Display **summary details of all cards in order and include the average value** as part of that summary.

    Sample output:

```
Sort by average...
Summary Data
1)
Spider-Man / Peter Parker (HERO)
sp: 80 | st: 75 | ag: 90 | in: 84
Average Stat: 82.25

2)
Spider-Man / Miles Morales (HERO)
sp: 82 | st: 77 | ag: 90 | in: 70
Average Stat: 79.75
```

Complete as much as you can within the time available balancing your efforts across the 3 sections. When finished, compress (zip) the entire **Eclipse solution** and upload to Assignments (P3 assessment) on CANVAS

Now : check the uploads to ensure you have submitted the correct files

[END]