CS3330: Lab 6

Due 48 hours after your lab ends

Objective:

- Understand and implement DatabaseDriver and Goon classes
- Learn the **instanceof** keyword to determine Object types at run-time

Submission Information:

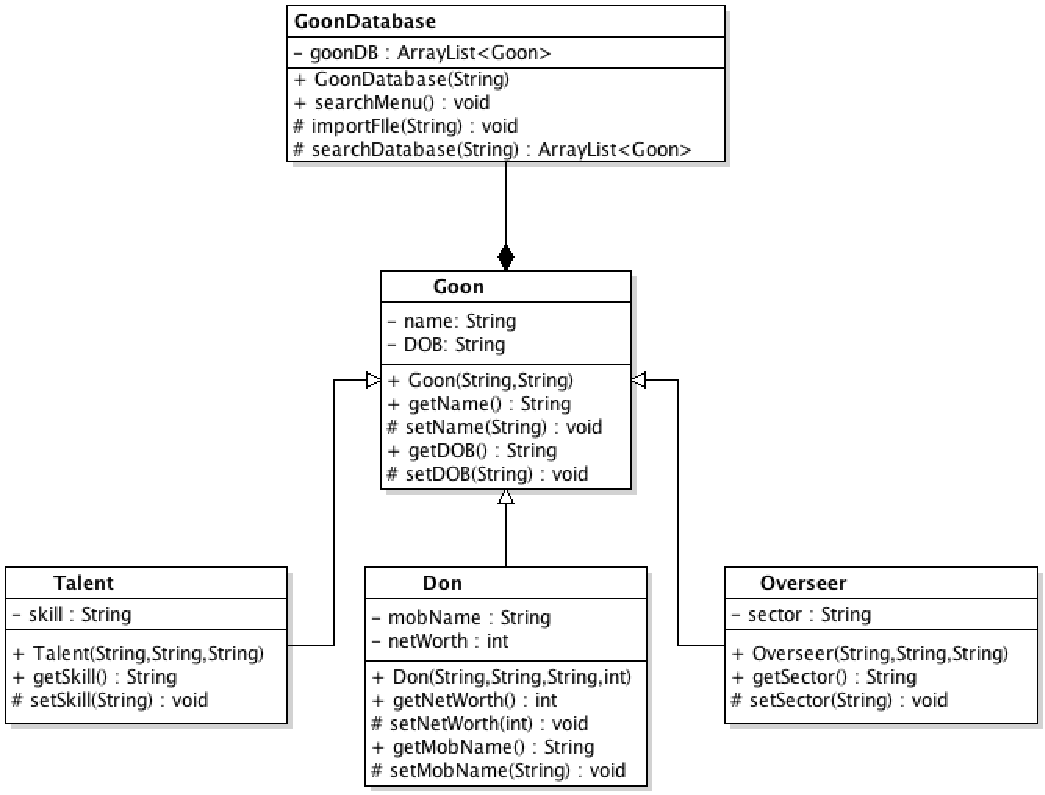
cs_submit CS3330_LAB-<section_letter> LAB6 <yourpawprint>.cs3330.lab6.zip

Purpose:

For this lab, you will be working with an ArrayList. You will create the classes according to the UML diagram and also construct a driver class. The aim of this lab is to construct objects based on file input and store them into an ArrayList. Once that is finished, the user will be able to search the database that was built and view stored information.

This lab consists of 6 different classes and a data file: DatabaseDriver.java, GoonDatabase.java, Goon.java, Talent.java, Overseer.java, Don.java, and Goons.csv

UML:



Additional Info:

Getters and setters are present for every attribute. Getters are public and Setters are private.

DatabaseDriver.java:
Your main method will instantiate a GoonDatabase object and call the searchMenu method.

GoonDatabase.java:
This will be the database wrapper. The constructor will accept the name of the file to be read in and call the importFile method. File located at Data/Goons.csv. You will need to create a Folder in your project called Data.

The searchMenu method will be a loop wherein the user will enter a query (or q to exit), the database will be searched with the searchDatabase method, and the results will be presented.

The searchDatabase method will accept a query string and search the ArrayList to see if the object contains the query (your search should be case-insensitive!) as the name or is of that object type (Hint: use **instanceof** to check for object type). It will return an ArrayList containing all of the values whose objects contain the query string. You **must** list the complexity of the search method in a comment next to the method header.

Goon.java:
This is the superclass it consists of a name and a date of birth of a Goon.

Talent.java:
Talents are going to be the bottom tier Goon. Each has a specific skill.

Overseer.java:
This is the mid tier Goon. Because of their skill, they are considered the master of a certain sector.

Don.java:
This is the top tier Goon, and is the leader of their mob. As the leader of such an organization, they are very rich.

Goons.csv:
This file contains information on the three different goon types. Each line will begin with the type of goon. Talents will list their name, date of birth, and their talent. Overseers will list their name, date of birth, and their sector. Dons will list their name, date of birth, the name of their mob, and their net worth.

Output Expected:

```
Import Complete.

Please enter your query (q to exit): Talent
Results:
1 - Jenny 'From the Block' Sanders
2 - Roxxy
3 - Rat
4 - Matt
5 - Jeremy
6 - Ankil

Please enter your query (q to exit): Ankil
Results:
1 - Ankil
Please enter your query (q to exit): overseer
Results:
1 - Vinny
2 - James
3 - 'Princess' Peach

Please enter your query (q to exit): james
Results:
1 - James

Please enter your query (q to exit): sand
No results found…

Please enter your query (q to exit): don
Results:
1 - Mario

Please enter your query (q to exit): q

Program Complete.
```

Grading:

General:

If your program does not compile, has runtime errors, or doesn't produce any I/O; your lab will have a grade of zero.

Not Having a Submission Code in each of the file you create will also result in a grade of ZERO.
You also may not create ANY other instance variables or methods.
You will LOSE 10 POINTS for doing either of these.

Rubric:

5 points – Comments (ie. Javadoc comments and inline comments)
6 points – For Constructors, Accessors, and Mutators in each of the classes
4 points – For listing the correct complexity of the search method
6 points – For following the UML exactly
4 points – For correctly implementing the search method
5 points – For correctly creating and populating the ArrayList and matching the output format 1-to-1

Bonus

Replace all usage of the ArrayList Object with a TreeMap Object, the key for each object are the occupation and name of the Goon. Also, in your header comments in DatabaseDriver.java, explain why a TreeMap is better suited than an ArrayList in some cases.