CS3330 LAB 3, DUE 48 HOURS AFTER YOUR LAB ENDS

Objectives:

- More practice with constructors, getters, and setters
- UML Class Diagrams to Java Classes
- Using the Random class

Submission Info:

Use the online submission system available at https://submit.cs.missouri.edu/app

Lab Material:

Download the main method, and Heroes.csv given with this LAB 3 document on blackboard. Please copy the contents from the main code given into a LabThreeDriver class; you create. Create a folder called DataFiles at the same level as your source folder (src). Example below:

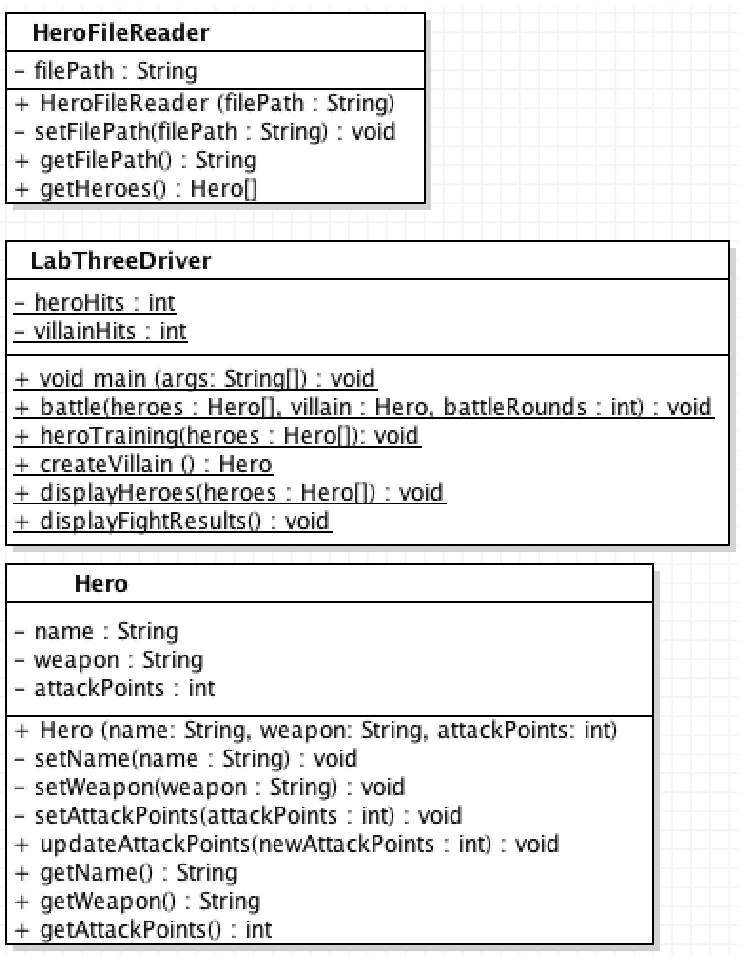
mypawprint.cs3330.lab3

    src/mypawprint/cs3330/lab3/.java files
    DataFiles/Heroes.csv

General Lab Info:

You will create 3 separate java classes in a project that you create.

UML CLASS DIAGRAMS:



```
HeroFileReader
- filePath : String
+ HeroFileReader (filePath : String)
- setFilePath(filePath : String) : void
+ getFilePath() : String
+ getHeroes() : Hero[]
```

```
LabThreeDriver
- heroHits : int
- villainHits : int
+ void main (args: String[]) : void
+ battle(heroes : Hero[], villain : Hero, battleRounds : int) : void
+ heroTraining(heroes : Hero[]): void
+ createVillain () : Hero
+ displayHeroes(heroes : Hero[]) : void
+ displayFightResults() : void
```

```
Hero
- name : String
- weapon : String
- attackPoints : int
+ Hero (name: String, weapon: String, attackPoints: int)
- setName(name : String) : void
- setWeapon(weapon : String) : void
- setAttackPoints(attackPoints : int) : void
+ updateAttackPoints(newAttackPoints : int) : void
+ getName() : String
+ getWeapon() : String
+ getAttackPoints() : int
```

CLASSES TO IMPLEMENT

LabThreeDriver.java

Attribute(s)

heroHits
villainHits

Method(s)

Type the following in for the main method:

```
public static void main(String[] args) {
       HeroFileReader heroFileReader = new HeroFileReader("DataFiles/Heroes.csv");
       Hero[] heroes = heroFileReader.getHeroes();
       LabThreeDriver.displayHeroes(heroes);
       Hero villain = LabThreeDriver.createVillain();
       System.out.println("Villain: " + villain.getName());
       System.out.println("");
       LabThreeDriver.battle(heroes,villain,10);
       LabThreeDriver.displayFightResults();
       LabThreeDriver.heroTraining(heroes);
       LabThreeDriver.heroHits = 0;
       LabThreeDriver.villainHits = 0;
       battle(heroes,villain,10);
       LabThreeDriver.displayFightResults();

}
```

displayHeroes(Hero[] heroes): displays the name of each hero from the passed parameter heroes. (Check the output for formatting)
createVillain(): returns an instance of Hero with the parameters of name = “Loki”, weapon = “Staff”, and attackPoints = 20.
heroTraining(Hero[] heroes): Updates each hero from the passed parameter heroes’ attackPoints by a multiplier of 2.
battle(Hero[] heroes, Hero villain, int battleRounds): Creates an instance of the Random class calling the anonymous constructor. Next, loops a counter until the value of the passed parameter battleRounds. Inside the loop, battle happens between a random hero vs the villain. To pick a hero to battle the villain, pick a random integer between 0 and the size of the heroes array. Next, pick another random integer between 0 and the picked hero’s attackPoints. If the randomly picked integer (random value picked randomly based off the hero’s attackPoints) is **greater than** the villain’s attackPoints, add one to the heroHits and display the battling hero’s name with the message “got a successful hit with” and the current battling hero’s weapon, else if the villain’s attackPoints is **greater than** the current hero’s attackPoints then add one to villainHits and display the villain’s name with the message “got a successful hit with” and the villain’s weapon.
displayFightResults(): displays the the current values of heroHits and number of villainHits. (Once again, refer to the output for formatting)

<u>Hero.java</u>

Attribute(s)

name
weapon
attackPoints

Constructor(s)

Hero(String name, String weapon, int attackPoints): Calls the setters to set the attributes name, weapon, and attackPoints.

Method(s)

setName(String name): sets the attribute name to the passed attribute name.
setWeapon(String weapon): sets the attribute weapon to the passed attribute weapon.
setAttackPoints(int attackPoints): Verifies that the attackPoint is positive (greater than 0) before setting it. Otherwise, set it to 10.
updateAttackPoints(): calls the setAttackPoints method with the updated attack points.
getName(): returns the name attribute.
getWeapon(): returns the weapon attribute.
getAttackPoints(): returns the attackPoints attribute.

<u>HeroFileReader.java</u>

Attribute(s)

filePath

Constructor(s)

HeroFileReader(String filePath): Calls the setter to set the attribute filePath.

Method(s)

setFilePath(String filePath): sets the attribute filePath to the passed parameter filePath.
getFilePath(): returns the attribute filePath.
getHeroes(): Near identical to Lab 2 process of reading a file except with a new Object type. Create an array of Hero that is size **5** that will be populated with data from the file located at the attribute filePath. The file structure is in the format of CSV (Comma Separated Values) with the name, weapon, and attackPoints in that respective order. Read the file line by line, and for each element in the new array of type Hero, create a new instance of Hero with the parsed out values.
Note: Never assume the file will be equal to your array size. Never assume the input will be nicely formatted. Watch out for empty lines read from the file.

************ SAMPLE PROGRAM OUTPUT *(HITS and HIT COUNTS WILL BE RANDOM)* ************

```
Hero: Captain America
Hero: Wolverine
Hero: Deadpool
Hero: Spider Man
Hero: Hulk

Villain: Loki

Villian Loki got a successful hits with his Staff
Deadpool got a successful hit with his Blades
Villian Loki got a successful hits with his Staff

Battle Stats
Hero Hits on Villian 1
Villain Hits on Heroes 2

Deadpool got a successful hit with his Blades
Hulk got a successful hit with his Fist
Hulk got a successful hit with his Fist
Deadpool got a successful hit with his Blades

Battle Stats
Hero Hits on Villian 4
Villain Hits on Heroes 0
```

GRADE GUIDE

30 possible points with a possible 5 bonus points extra

If your program does not compile, produce any input/output (I/O) because most of the source code is commented out then your lab will receive a grade of zero points. If your lab has any runtime errors (Such as NullPointerException or ArrayIndexOutOfBounds), the lab will also receive zero points. **If you don't have header comments inside of all your class files, you will receive a zero points as well. NO EXCEPTION!!!!!**

Grading Rubric

5 points: Javadoc and commenting
5 points: HeroFileReader class
6 points: Hero class
6 points: battle method
4 points: displayFightResults and createVillain methods
4 points: heroTraining method

BONUS (5 points - ALL OR NOTHING)

Convert all Hero[] arrays in your program into ArrayList<Hero> and make any change that will allow the data structure ArrayList<Hero> to work properly. Updating HeroDataReader.java and LabThreeDriver.java is required. Output is still exact same as the normal lab.