CS 3560 – Assignment #2

Maximum Points: 100 pts.

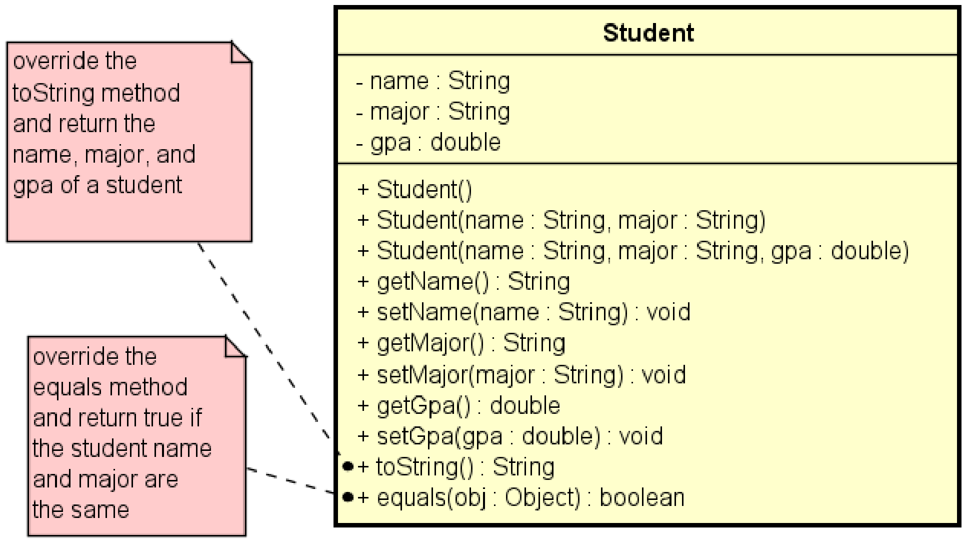
Bronco ID: |__|__|__|__|__|__|__|__|__|

Last Name: ______________________

First Name: ______________________

Note 1: Your submission header must have the format as shown in the above-enclosed rounded rectangle.
Note 2: Homework is to be done individually. You may discuss the homework problems with your fellow students, but you are NOT allowed to copy – either in part or in whole – anyone else's answers.
Note 3: All submitted materials must be legible. Figures/diagrams must follow the given instructions.
Note 4: Your deliverable should be a .pdf file submitted through Gradescope by the deadline. Do not forget to assign a page to each of your answers when making a submission. In addition, source code (.java files) should be added to an online repository (e.g., GitHub) to be downloaded and executed later.
Note 5: Please use and check the Canvas discussion for further instructions, questions, answers, and hints. The bold words/sentences provide information for a complete or accurate answer.

1. [15 points] Given the corresponding UML class below, answer the following questions.


override the toString method and return the name, major, and gpa of a student
override the equals method and return true if the student name and major are the same
Student
- name : String
- major : String
- gpa : double
+ Student()
+ Student(name : String, major : String)
+ Student(name : String, major : String, gpa : double)
+ getName() : String
+ setName(name : String) : void
+ getMajor() : String
+ setMajor(major : String) : void
+ getGpa() : double
+ setGpa(gpa : double) : void
+ toString() : String
+ equals(obj : Object) : boolean

   a) [10 points] Write the corresponding Java code for the `Student` class. Make sure to include the **expected code** inside the **methods** and **constructor(s)** when appropriate.
   b) [5 points] Write a driver class called `StudentTest` that instantiates 2 students inside its main() method. The first student {"John", "CS", 3.5} is created by using a **parameterized** constructor while the second student {"Mary Ann", "CE", 3.3} is created by using the **no-arg constructor**. After creating both students and updating their name, major, and gpa, **print** their states by using the `toString()` method.

2. [15 points] Given the Java code below, answer the following questions.

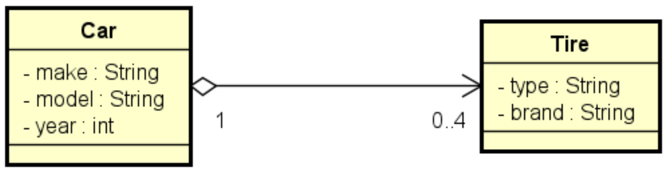
```
public class Box {
   private double length, width, height;
   private static int numberBoxes;
   public Box(double length, double width, double height) {
      this.length = length;
      this.width = width;
      this.height = height;
      numberBoxes ++;
   }
   public double volume() {
      return (length * width * height);
   }
   public double surfaceArea() {
      return ((2 * length * width) + (2 * length * height) + (2 *
                                                     width * height));
   }
}
```

a) [10 points] Draw the corresponding UML class diagram **(astah must be used)**. Add some notes to explain what the `Box()`, `volume()`, and `surfaceArea()` methods should do/return.
b) [5 points] Explain the **practical implications** of making `length`, `width`, and `height` as instance fields while `numberBoxes` is defined as a static field.

3. [10 points] Draw the UML class diagrams for the proposed scenarios below **(astah must be used)**. Include the attributes shown in the {} inside your classes by choosing appropriate data types. No methods need to be included.

a) [5 points] A given actor {name, gender} can act in multiple movies {name, genre}, and for each movie, we need to register the **multiple characters** {role} played by the actor.
b) [5 points] An order {id, customer, date} can have multiple products {name, price} and for each product, we need to register its **details** {quantity} in the order.

4. [10 points] Based on the UML class diagram below, draw the appropriate object diagrams that will correspond to the proposed system states. **Include all attributes** inside your objects.

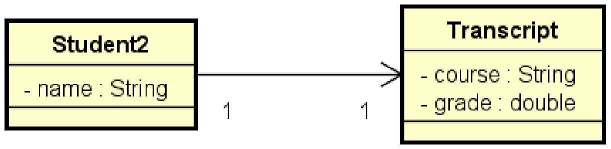


a) [5 points] **One** tire has been **created** {Summer, Goodyear} and **added** to the car {Kia, Spectra, 2006}.

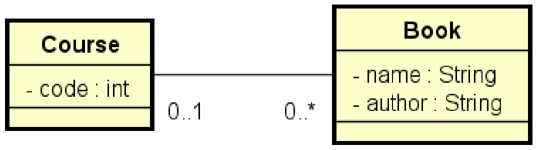
b) [5 points] **Four** tires have been **created** {Summer, Goodyear}, {Summer, Goodyear}, {Summer, Firestone}, and {Summer, Firestone} but **only** the **Goodyear's** ones have been added to the car {Kia, Spectra, 2006}.

5. [50 points] Based on the relationships shown in the UML class diagrams below, write the corresponding Java code. Include all **fields**, **constructors**, and **other extra methods** as guided by the instructions (getters and setters not needed). Also, include the expected behavior of each of those methods.

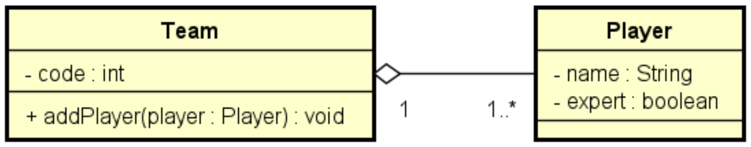
a) [5 points]. Student and Transcript: fully parameterized constructors.



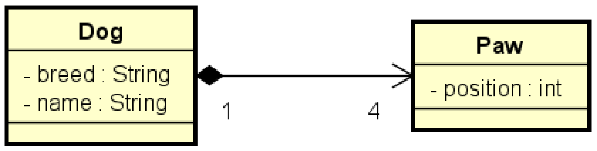
b) [5 points]. Course: fully parameterized constructor. Book: no-arg constructor.



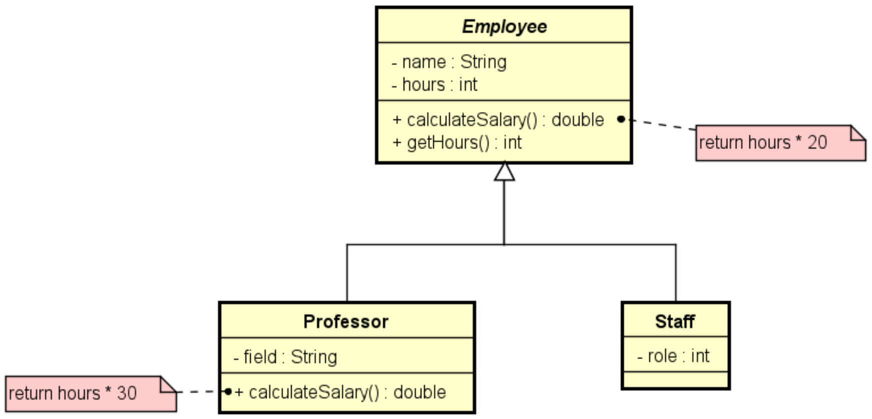
c) [5 points]. Team: no-arg constructor. Player: fully parameterized constructor.



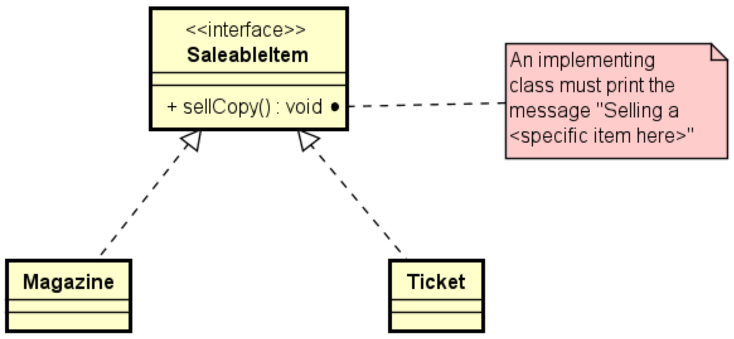
d) [5 points]. Dog: no-arg constructor. Paw: fully parameterized constructor.



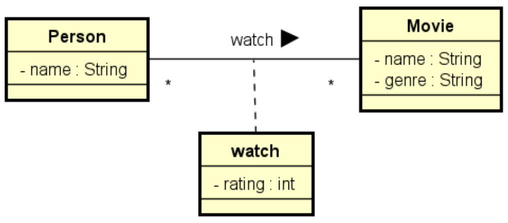
e) [8 points]. Employee, Professor, and Staff: fully parameterized constructors.



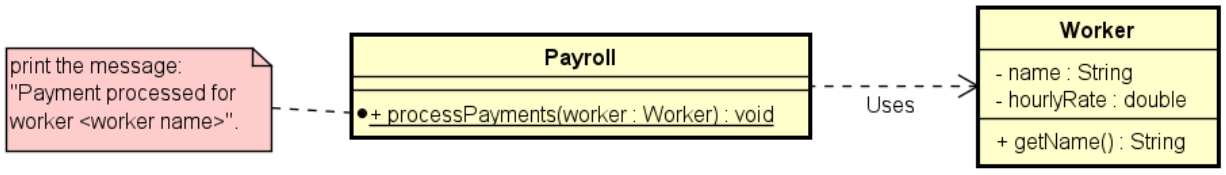
f) [8 points]. Magazine and Ticket: no-arg constructors.


<<interface>>
SaleableItem
+ sellCopy() : void
An implementing class must print the message "Selling a <specific item here>"
Magazine
Ticket

g) [8 points]. Person, Movie, and Watch: fully parameterized constructors.



h) [6 points]. Payroll: no-arg constructor. Worker: fully parameterized constructor.


print the message: "Payment processed for worker <worker name>".
Payroll
+ processPayments(worker : Worker) : void
Uses
Worker
- name : String
- hourlyRate : double
+ getName() : String

Important Note: Answers to all questions should be written clearly, concisely, and unmistakably delineated. You may resubmit multiple times until the deadline (the last submission will be considered).

NO LATE ASSIGNMENTS WILL BE ACCEPTED. ALWAYS SUBMIT WHATEVER YOU HAVE COMPLETED FOR PARTIAL CREDIT BEFORE THE DEADLINE!