Include your full name and your partner’s full name at the top of each file, using a Java Javadoc   
/\*\* @author \*/ tag (for example: /\*\* @author Paul Mills and Jason Wilder \*/).

**Session 7 Lab: Designing a sports league program**

In this lab you will draft an outline for a program that supports players, teams and a league. It doesn’t matter what the sport or league is (Hockey, Soccer, Baseball, Curling, etc.) You decide. It must be a sport where each team consists of multiple players (no less than five) multiple teams, and is one league.

We are practicing our design skills here doing some of the preparation needed before any code is written. Take the mindset that you will be coming back to a customer with a proposal.

**Questions you must consider:**

* For players:
  1. What attributes does a player have? First name, last name, age, weight, position, shirt number, shoes size, dominant foot, goals.
  2. What fields would be required? First name, last name, age and weight.
  3. What constructors are required to create players? Two. One including the shirt number and one without it. Since it can be assigned after.
  4. What validations are required? Valid name, valid age, valid weight, valid position, valid shoes size, valid dominant foot.
  5. What functions does a player perform? Run, pass the ball, score a goal, tackle another player.
* For teams:
  1. What attributes does a team have? Name, players
  2. What fields would be required? Name, number of players.
  3. What constructors are required to create teams? One with required fields.
  4. What validations are required? Valid name.
  5. What functions does a team perform? Transfer a player, sign a new player, play a match.
* For the league:
  1. What attributes does a league have? Name, number of teams, category.
  2. What fields would be required? Name, number of teams, category.
  3. What constructors are required to create a league? One.
  4. What validations are required? Valid name, valid number of teams, valid category.
  5. What functions does a league perform? Add a team to the league, promote a team, relegate a team.

In designing your program you need to consider the topics we covered today, which include RDD, Cohesion, Coupling, Duplication, etc..

**Additional things you should consider:**

* What are the relationships between the three classes? How do teams track players and how does the league track teams? A team has a list of players and a league has a list of teams.
* Is the design reusable? Yes.
* Is the design extendable? Yes.
* Clarity. It should be clear to the Customer what you have designed. Remember, they won’t be programmers so you have to be reasonably detailed in your descriptions and explanations.

**Deliverable:**

* A text-based document that clearly breaks things down class by class. Bullet points are fine. You’re writing a program outline not an essay.
* This exercise is about thinking things through before you write the code
* Write a simple jUnit test for some of the class features, and pass your own tests.
* Do this lab with your partner. Create the identical code as your partner, and show your instructor before the start of the next class. Keep these papers and bring them to the final exam, to get your marks:
* Instructor Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Player Class**

* Attributes:
  + First name
  + Last name
  + Age
  + Weight
  + Position
  + Shirt number
  + Shoes size
  + Dominant foot
  + Goals
  + Assists
* Fields:
  + **String firstName**
  + **String lastName**
  + **int age**
  + **double weight**
  + **String position**
  + **int shirtNumber**
  + **int shoesSize**
  + **String dominantFoot**
  + **int goals, assists.**
* Constructors:
  + One with parameters all parameters except assists, goal and shirtNumber.
  + Another with all the parameters.
* Validations:
  + Valid name (not null or empty for first and last name).
  + Valid age (positive integer).
  + Valid weight (positive double).
* Functions:
  + **run()**
  + **passBall()**
  + **scoreGoal()**
  + **tacklePlayer()**
  + **assistGoal()**

**Team Class**

* Attributes:
  + Name
  + List of players
* Fields:
  + **String name**
  + **ArrayList players**
* Constructors:
  + One with parameters for name.
* Validations:
  + Valid name (not null or empty).
* Functions:
  + **transferPlayer(Player player, Team otherTeam)**
  + **signNewPlayer(Player player)**
  + **playMatch()**

**League Class**

* Attributes:
  + Name
  + List of teams
  + Category
* Fields:
  + **String name**
  + **int numberOfTeams**
  + **String category**
* Constructors:
  + One with parameters for name and category.
* Validations:
  + Valid name (not null or empty).
  + Valid category (not null or empty).
* Functions:
  + **addTeamToLeague(Team team)**
  + **promoteTeam(Team team)**
  + **relegateTeam(Team team)**