/\*\* @author Wasi Omar Bin Bayzed\*/

**Player Class:**

The class and file names are both Player.

**Attributes:**

- Player ID

- Name

- Age

- Position

- Skill Level

- Team ID (to track the team the player belongs to)

**Fields:**

- playerId (int)

- playerName (String)

- playerAge (int)

- playerPosition (String)

- skillLevel (int)

- teamId (int)

**Constructors:**

- Default constructor

- Parameterized constructor to initialize all fields

**Validations:**

- Ensure age is a positive integer

- Skill level is within a valid range

- Player name is not empty

**Functions:**

- Getters and setters for all attributes

- displayPlayerInfo() method to display player information

**Team Class:**

The class and file names are both Team.

**Attributes:**

- Team ID

- Team Name

- List of Players (ArrayList or other suitable collection)

**Fields:**

- teamId (int)

- teamName (String)

- players (ArrayList<Player>)

**Constructors:**

- Default constructor

- Parameterized constructor to initialize teamId and teamName

**Validations:**

- Team name is not empty

- Ensure the team has at least 5 players

**Functions:**

- Add player to the team

- Remove player from the team

- Getters and setters for attributes

- displayTeamInfo() method to display team information

**League Class:**

The class and file names are both League.

**Attributes:**

- League Name

- List of Teams (ArrayList or other suitable collection)

**Fields:**

- leagueName (String)

- teams (ArrayList<Team>)

**Constructors:**

- Default constructor

- Parameterized constructor to initialize leagueName

**Validations:**

- League name is not empty

- Ensure the league has at least two teams

**Functions:**

- Add team to the league

- Remove team from the league

- Display league standings (optional)

- Getters and setters for attributes

- toString() method to display league information

***Relationships:***

Between Player and Team:

- A player belongs to one team. Use the player's teamId to establish the relationship.

Between Team and League:

- A team belongs to one league. Use the team's leagueId or a direct reference to the league.

***Reusability and Extendability:***

Design Reusability:

- The Player, Team, and League classes are designed for reuse in different sports leagues.

Design Extendability:

- Easily extendable to add more attributes or functionalities for future requirements.

***Clarity:***

Documentation:

- Clear, concise comments in the code to explain each class, attribute, and method.

Communication:

- Provide clear explanations to the customer about the designed classes, their attributes, relationships, and functionalities.