

# Java Method Overriding

When a subclass inherits from a superclass, it also inherits its methods; however, it can also *override* the superclass methods (as well as declare and implement new ones). Consider the following *Sports* class:

```
class Sports{
    String getName(){
        return "Generic Sports";
    }
    void getNumberOfTeamMembers(){
        System.out.println( "Each team has n players in " + getName() );
    }
}
```

Next, we create a *Soccer* class that inherits from the *Sports* class. We can override the *getName* method and return a different, subclass-specific string:

```
class Soccer extends Sports{
    @Override
    String getName(){
        return "Soccer Class";
    }
}
```

**Note:** When overriding a method, you should precede it with the `@Override` annotation. The parameter(s) and return type of an overridden method must be exactly the same as those of the method inherited from the supertype.

## Task

Complete the code in your editor by writing an overridden *getNumberOfTeamMembers* method that prints the same statement as the superclass' *getNumberOfTeamMembers* method, except that it replaces *n* with **11** (the number of players on a Soccer team).

## Output Format

When executed, your completed code should print the following:

```
Generic Sports
Each team has n players in Generic Sports
Soccer Class
Each team has 11 players in Soccer Class
```