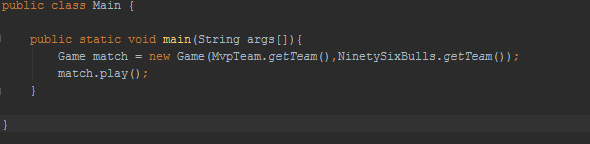
**Final Project Overview**

**Nicholas Solano**

**Comp 155**

**Main.java**

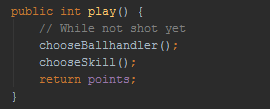


* Here in the main function of my game, I created a new object of the **game** class and named it **match** that I had created and called the function **play** so I can create an instance of the game happening.
* I then call my class of Players from each team and load them by using the function .getTeam(). This way it is more efficient because we can even create more teams for different match ups later in the future, by creating different instances of the .play().

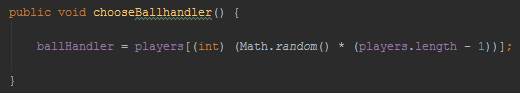
**Team.java**



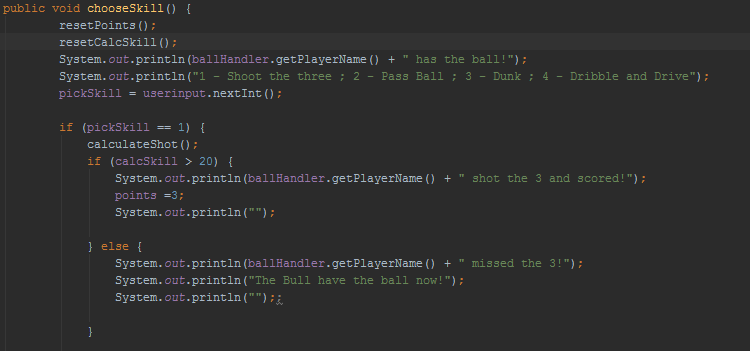
* For Team’s I create a array of objects, because for each player in game is an object, so in order to store them into an array we needed to make an array of objects.

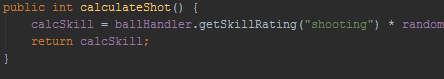


* For the function play, I called some methods I created to create a way in which the user could play the game. I then return the value that is set to points inside of chooseSkill() where it is then added to the score in the function Game.java.

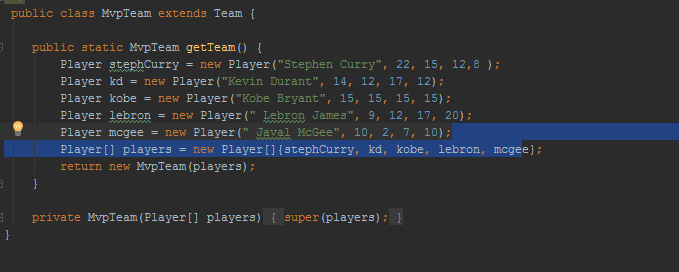


* Here I assign the object Player Ballhandler to a random player inside the array and it randomly selects an object in the array.



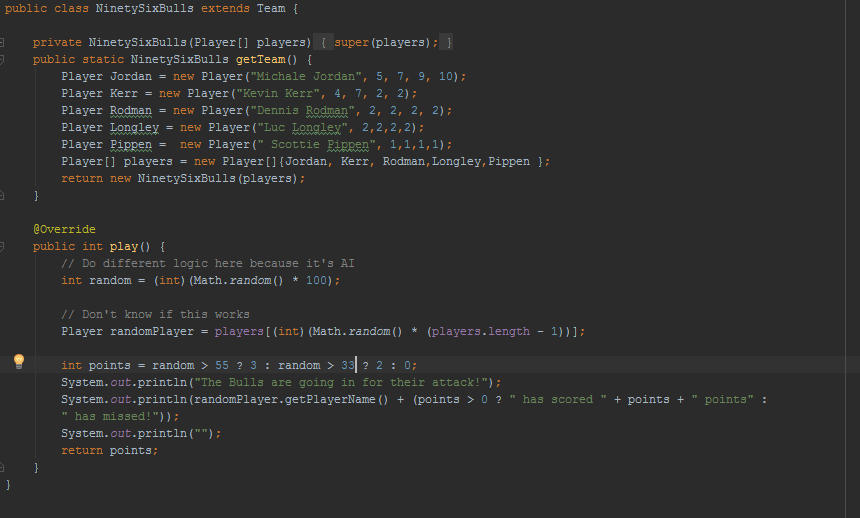
* For chooseSkill the reason I have the functions, resetPoints() and resetCalcSkill() is because I need to reset those values back to 0 or else it could cause an error by carrying misinformation from a previous instance for the next time we create an instance of play() in our Game.java file. I then created a method of if and else statements that are generally the same for all four options of the player.
* So if pickSkill is == 1 then it will use the function calculateShot(),
* Where calculateShot() generates a random number and multiplies it by the current objects rating in “shooting” and compares it to the value 20. If the calcSkill is > 20 then the character scores a point, if not he turns over the basketball to the other team.
* This happens for each case, so for Shooting, Passing, Layup, and Dunking it all works the same as this first if statement.

**MvpTeam.java**



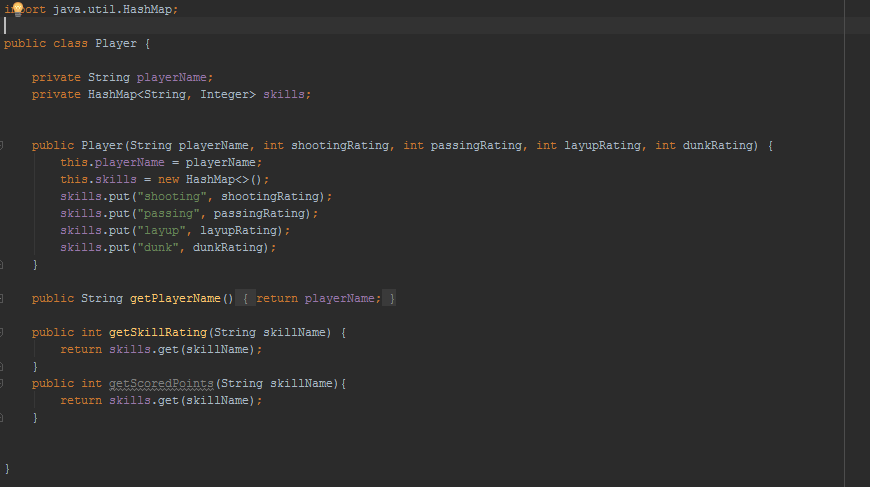
* So here we have a function get team, that stores each object into an array and returns the new array **players** back to the Team function so the program can know what the array is made of.

**NinetySixBull.java**



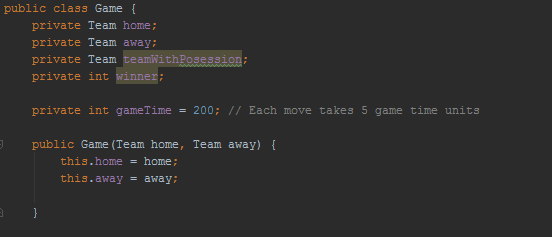
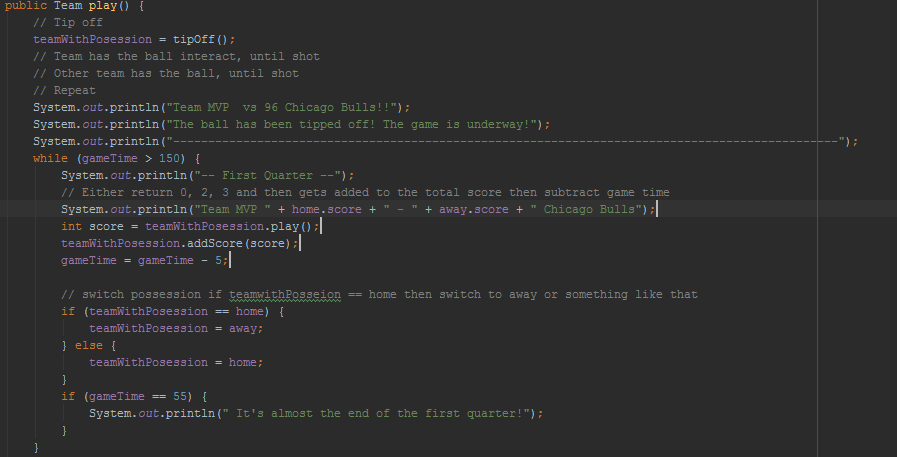
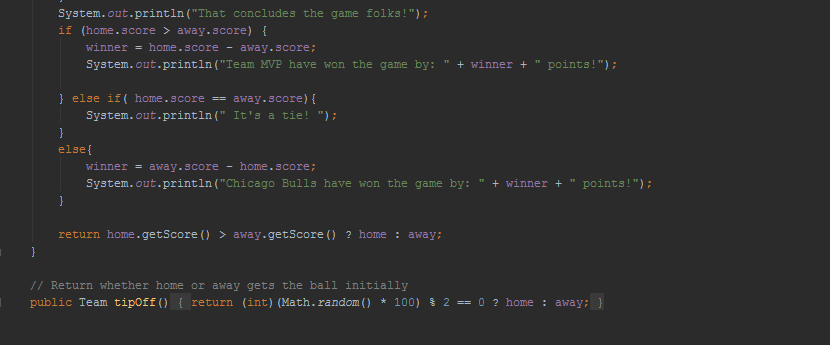
* This class works the same in the first part as MvpTeam, it just has different objects stored in the array.
* I actually override the play() function inside of the Team class and created a general option where I can sort of have an basic AI that plays for the other team.
* So what happens here is I create a random value from 1-100 and then I say if that random value is >55 then points will = 3 if random >33 then points will =2 otherwise points = 0.
* I thought this part was super neat and was kind of happy when it worked in the end!

**Player.java**



* So for this class what I did was I created a Map called skills, where it had the properties o both a String and integer. The reason I used a HashMap was because it is easier, and more efficient to access the different attributes I was going to give to each object, or character so that I could multiply those values and have each character have its own specific set of abilities..
* This way when I created an object of players, it was easier for me to organize and access the information rather than iterating through each object trying to find the name of that object and picking each value at each spot… it would be much more difficult to do so, so this is why I used a Map to make things simpler for the developer.

**Game.java**

* For Game.java this is where I created an object, that inside of it has a method called play() also that creates sort of the skeleton of how the game should be played in general.
* 
* So first things first, I created two objects for each team and then in the Main class I created those objects.
* And I also have a tipOff() function that determines which team actually starts with the ball at the beginning of the game… That is why I have a object of Team called teamWithPosession. That way we can determine which team is in possession of the ball at the given time, so we can alternate between teams going on offense and defense.
* Since actual time would be difficult to calculate. I made it so the total game time was 200 , where for each team would get a total of 5 offensive turns each quarter.
* 
* In the play function I have a total of 4 while loops, each one have a condition where it’s either 150, 100, 50 or 0. The reason I chose these is because of each quarter both teams would get a shot 5 times at offense.
* In the while loop I want to use the teamWithPossession and use the play() function inside of my Team class and allow the user to play as their character that is randomly chosen. That .play() will return a value of 0, 2 or 3 and assign it the variable score. I then take that score and add it to the team that is in possession of the ball’s total score and subtract 5 from the game clock to make sure we can go into the next quarters.
* 
* The final touch to the game happens where I take the scores and compare them to determine the winner of the game. I also list how much the winner won by.
* At the bottom here I have my tip off function that says If a number between 1-100 remainder 2 is == 0 then give the ball to home, if not give it to the away team for starting possession.

Overall this game was fun to make, it took me a while to learn how to do all the stuff, but I really enjoyed doing this project. I hope this project was beyond satisfactory to what you were looking for, it really took a lot of thinking to get it all worked together. I appreciate you teaching me this year, and hope to see you in future classes KJ!

Thanks,

Nicholas Solano

300136694