Team Panda APCS1 pd02
Tim Marder, Ryan Kim, Sean Roudnitsky 01/05/2018

Pandamonium Football UML

- Class *Football*
  - Instance Variables:
    - **String name** - the name of the user but will display as "Coach <name>".
    - **String team** - the team that the user chooses to coach.
    - **String menu** - the coach's dashboard for managing the team.
  - Methods:
    - **String usersName** - asks the user for their name and then returns "Coach <name>".
    - **String chooseTeam** - asks the user what team they would like to coach by returning a string of the list of teams.
    - **String teamChosen** - returns the welcome message.
    - **String mainMenu** - returns the "dashboard" for the coach along with information such as the team's name and overall team rating.
- Class *Players*
  - Instance Variables:
    - **int teamRating** - the overall rating of the entire team. (Average of players' ratings)
  - Methods:
    - **String[] listAlphabetical** - list the players in alphabetical order.
    - **String[] listRating** - list the players in order of ratings.
    - **String[] showStats -** shows the statistics for an individual player.
    - **int setRating** - averages the statistics of a player and gets the rating for that player.
    - **int setTeamRating** - averages the ratings of the players to get the overall rating of the team.
- Class *OffensivePlayers*
  - Instance Variables:
    - **String[] yourOffensivePlayers** - the offensive players that you own.
    - **String[] draftableOffensivePlayers** - the offensive players that you can draft.
  - Methods:
    - **String[] setYourOffensivePlayers** - returns an array of the offensive players that you own.
    - **String[] setDraftableOffensivePlayers** - sets the offensive players that you can draft.

        - **String toString** - to be able to list the elements of the arrays.

- Class *DefensivePlayers*
    - Instance Variables:
        - **String[] yourDefensivePlayers** - the defensive players that you own.
        - **String[] draftableDefensivePlayers** - the defensive players that you can draft.
    - Methods:
        - **String[] setYourDefensivePlayers** - returns an array of the defensive players that you own.
        - **String[] setDraftableDefensivePlayers** - sets the defensive players that you can draft.
        - **String toString** - to be able to list the elements of the arrays.
- Class *Quarterback*
    - Instance Variables:
        - **String position** - holds the position of the player.
    - Methods:
        - **String[][] setStats** - will hold the stats for each skill for every player. (Skills differ for every position).
- Class *Receiver*
    - Instance Variables:
        - **String position** - holds the position of the player.
    - Methods:
        - **String[][] setStats** - will hold the stats for each skill for every player. (Skills differ for every position).
- Class *Runningback*
    - Instance Variables:
        - **String position** - holds the position of the player.
    - Methods:
        - **String[][] setStats** - will hold the stats for each skill for every player. (Skills differ for every position).
- Class *Linebacker*
    - Instance Variables:
        - **String position** - holds the position of the player.
    - Methods:
        - **String[][] setStats** - will hold the stats for each skill for every player. (Skills differ for every position).

- Class *Cornerback*
  - Instance Variables:
    - **String position** - holds the position of the player.
  - Methods:
    - **String[][] setStats** - will hold the stats for each skill for every player. (Skills differ for every position).
- Class *Safety*
  - Instance Variables:
    - **String position** - holds the position of the player.
  - Methods:
    - **String[][] setStats** - will hold the stats for each skill for every player. (Skills differ for every position).
- Class *Shop*
  - Instance Variables:
    - **String shop -** the shop menu.
  - Methods:
    - **String buy -** updates the team's budget and returns a successful buy message.
- Class *Helmet*
  - Instance Variables:
    - **String[] helmets -** stores all the helmets in the shop.
  - Methods:
    - **void setHelmets** - updates the helmets available to buy once they were purchased from the shop.
    - **void riddel -** updates the stats of the team.
    - **void schutt** - updates the stats of the team.
    - **void xenith -** updates the stats of the team.
- Class *Pads*
  - Instance Variables:
    - **String[] pads** - stores all the pads in the shop.
  - Methods:
    - **void setPads** - updates the pads available to buy once they were purchased from the shop.
    - **void underarmour -** updates the stats of the team.
    - **void adidas** - updates the stats of the team.
    - **void nike** - updates the stats of the team.

- Class *Cleats*
  - Instance Variables:
    - **String[] cleats** - stores all the cleats in the shop.
  - Methods:
    - **void setCleats** - updates the pads available to buy once they were purchased from the shop.
    - **void alphas -** updates the stats of the team.
    - **void highlights** - updates the stats of the team.
    - **void untouchables** - updates the stats of the team.
- Class *Game*
  - Instance Variables:
    - **String[] games** - all the games to play in the game menu.
    - **String lastGameWon** - the latest game that the user won.
    - **int yourScore** - the user's score.
    - **int theirScore** - the opponent's score.
    - **boolean wasOffense -** if the user was on offense for the current drive.
    - **boolean wasDefense** - if the user was on defense for the current drive.
    - **int quarter** - the quarter of the current game.
  - Methods:
    - **void playGame** - method to start the game.
    - **String coinToss** - allows the user to choose heads or tails.
    - **String winToss** - allows the user to choose whether to receive or kickoff the ball.
    - **String loseToss** - the AI chooses whether to receive or kickoff the ball.
    - **String win -** congratulates the user on winning. Adds value to the team's budget. Updates lastGameWon.
    - **String lose** - lets the player know they lost. Adds a smaller value to the team's budget.
    - **String tie** - lets the player know he tied with the opposing team. Adds the same amount to the team's budget as a win would get but does not update the lastGameWon.
    - **String superbowlWon** - if the user wins the superbowl, the game stops because that is the last "level" that the user has to beat.

- Class *Offense*
  - Instance Variables:
    - **int downCounter** - keeps track of what down it is.
    - **int yardsToGo** - keeps track of how many yards are left to convert 1st down.
  - Methods:
    - **String playOffense** - will put the user onto Offense.
    - **String run** - if the user chooses the run the ball. Updates the user with the amount of yards gained and how many yards to go.
    - **String shortPass** - if the user chooses to pass the ball short. Updates the user with the amount of yards gained and how many yards to go.
    - **String longPass** - if the user chooses to pass the ball long. Updates the user with the amount of yards gained and how many yards to go.
    - **String goodRun** - if the user gets lucky and gets a good run.
    - **String averageRun** - highest chance for the user to get an average amount of yards from the run.
    - **String badRun** - if the user gets unlucky and the runningback gets tackled for a loss of yards.
    - **String successfulPass** - when the user successfully completes the pass. Lets the user know how many yards gained and how many yards to go.
    - **String incompletePass** - when the user's pass gets either thrown poorly or swatted by the defense. Updates the down counter.
    - **String getIntercepted** - when the user gets unlucky and pass gets intercepted by the opposing team's defense. Puts the user on defense.
    - **String getSacked** - a chance that when the user chooses to pass, the defense might sack the quarterback. Will update the down counter.
    - **String firstdownConverted** - Updates the down counter back to 1 and lets the user know that he/she has reached first down.
    - **String failToConvert** - If the user fails to convert the 1st down, puts the user on defense and updates with the yards to go for the opponent.
    - **String touchdown** - If the user's offense scores, updates the score, and gives option whether to kick an extra point or do a two point conversion.
    - **String extraPoint** - If user chooses to kick extra point, a point is added if user scores.
    - **String twoPoint** - If user chooses to try a two point conversion, gives user an option whether to pass or run the ball. If user succeeds, two points are added to the score.

        - **String fieldGoal** - If user is at fourth down and close enough to the endzone (40 yards), an option is to kick a field goal for 3 points instead of risking by converting the 1st down or scoring a touchdown.
        - **String punt** - If user is at fourth down but not close enough to the endzone, an option is to punt the ball as far as possible so that when the opposing team is on offense, they have a longer distance to go back to score. This is a less risky option than to try to gain the first down.
- Class *Defense*
    - Instance Variables:
        - **int opponentDownCounter** - keeps track of the opponent's downs.
        - **int opponentYardsToGo** - keeps track of how many yards the opponent has to move up the ball to convert the 1st down.
    - Methods:
        - **String playDefense** - puts the user onto defense.
        - **String zoneCoverage** - If the user chooses to play zone coverage, has a higher chance of preventing the opponent to gain more yards.
        - **String manCoverage** - If the user chooses to play man coverage, there's now a higher chance of the defense letting a big play slip by but also has a higher chance of getting an interception.
        - **String blitz** - If the user chooses to blitz, it's the riskiest option for the defense to let a big play slip by but also a higher chance of sacking the opponent's quarterback.
        - **String tackle** - Will let the user know how many yards the opponent gained before tackled and how long they have to go till the endzone.
        - **String sack** - If the user's defense tackles the quarterback, it is a sack. Will update the down counter and cause the opposing team to lose yards.
        - **String missTackle** - If the user gets unlucky and one of the defensive players misses a tackle, it will cause a bigger play than usual for the opposing team, sometimes even a touchdown.
        - **String intercept** - If the user gets lucky, one of the defensive players will intercept the opposing team's ball when passed by them and try to run it back for as many yards as possible. Will put the user on offense.
        - **String tackleForLoss** - If the user gets lucky and the defense is able to read the play quickly and tackle the runningback behind the line of scrimmage for a loss of yards for the opposing team.
        - **String swat** - if the opposing team throws the ball and the receivers are being well guarded by the user's defense, one of the players will swat the ball away and cause an incomplete pass. Will update the down counter.

        - **String turnover** - if the opponent is on fourth down and punts the ball or tries to run/pass the ball to try to convert the 1st down but fails, causes a turnover and puts the user on offense.

- Class *SpecialTeams*
    - Instance Variables:
        - **int chance** - the chance of the special team player of making a successful play based on the difficulty of the opposing team.
    - Methods:
        - **String playSpecial** - puts the user onto special teams.
        - **String fieldGoal** - when the user attempts to kick a field goal on fourth down and when is close enough to the endzone. Will tell the user if he/she has succeeded and update the score. Will put the user on defense after that.
        - **String punt** - when the user is on fourth down and not close enough to the endzone and decides to punt the ball. Will tell the user the amount of yards the punt flew and how many yards the opposing team has run it back. Will put the user on defense after that.