Name: Om Patel

CodeHS

Soccer League Project

GUI.java

```
/******************************************************************************
* File name: Soccer League
*
* Description:
* A program where you can manage soccer teams.
* You can create teams, see who plays against whom each
* week, and check how well teams are doing based on wins,
* goals scored, and clean sheets. It's designed to let
* users easily organize and track simulated soccer league matches
* and team stats.
*
* Author: Om Patel
*
* Date: June 16 2024
*
* Concepts:
* Use of Swing components (JFrame, JButton, JLabel, JTextArea, JScrollPane, JOptionPane)
* Use 0f ActionListeners
* Use of ArrayList to store information about the players
* Use of Scanner
* Use of Random to shuffle between the players
* Use of try-catch to handle potential errors
*****************************************************************************/

import javax.swing.*;
import java.util.ArrayList;
import java.util.Random;
import java.util.Scanner;
import java.io.File;

/* Class representing the GUI for managing soccer teams and games. */
public class GUI {

    public static Game game = new Game();


   /* Creating ArrayList called teamNames, containing all the team names */
    public static JFrame mainFrame;
    public static String[] teamNames = {
        "Swift Stars ", "Blaze United", "Echo", "Crested Lions", "Horizon ",
        "Ember United", "Crest", "Radiant Tigers", "Peak United", "Volt", "Stellar",
        "Crested Eagles", "Nova", "Swift Tigers", "Apex United", "Dynamic", "Zenith", "Solar",
        "Lunar", "Eclipse"
    };

   public static ArrayList<Player> goalkeepers = new ArrayList<>();
   public static ArrayList<Player> defenders = new ArrayList<>();
   public static ArrayList<Player> midfielders = new ArrayList<>();
   public static ArrayList<Player> strikers = new ArrayList<>();
   public static ArrayList<Team> teams = new ArrayList<>(); /* List of the teams */
   public static ArrayList<String[]> allMatchups = new ArrayList<>(); /* List off all the Matchups */
   public static ArrayList<Game> currentWeeklyGames = new ArrayList<>(); /* List of all the Weekly games being played */
   public static int currentMatchupIndex = 0;

    public static void main(String[] args) {
        readPlayersFromFile("Player.txt"); /* Read player information from file */
        createMainGUI(); /* Create the GUI */
        createTeams(); /* Create teams using the players given */
        generateAllMatchups(); /* Generate all possible matchups */
    }
    /* Method to create the main GUI */
    public static void createMainGUI() {
        mainFrame = new JFrame("Soccer Teams");  /* Create a new JFrame for Soccer Team */
        mainFrame.setSize(500, 600);   /* Set size of the main frame */
        mainFrame.setLayout(null);  /* Set layout to null */

        /* Initialize vertical and horizontal positions for button placement */
        int vertical = 20;
        int horizontal = 20;

        /* Loop through teamNames array to create buttons */
        for (int i = 0; i < teamNames.length; i++) {
            JButton button = new JButton(teamNames[i]); /* Create a new button with team name */

            if (i < 10) {     /* Check if button index is less than 10 */
                button.setBounds(50, vertical, 200, 30); /* Set button position and size */
                vertical += 35; /* Increment vertical position for next button */
            } else {  /* If button index is 10 or greater */
```

```java
                button.setBounds(270, horizontal, 200, 30); /* Set button position and size */
                horizontal += 35; /* Increment horizontal position for next button */
            }

            int index = i; /* Store current index for action listener */

             /* Add action listener to button */
            button.addActionListener(e -> {
                if (!teams.isEmpty()) {  /* Check if teams list is not empty */
                    createConsole(teams.get(index)); /* Create console for selected team */
                }
            });

             mainFrame.add(button); /* Add button to mainFrame */
        }

        /* Create 'Create Teams' button */
        JButton createTeamsButton = new JButton("Create Teams");
        createTeamsButton.setBounds(10, 500, 150, 30); /* Set button position and size */
        /* Add action listener to 'Create Teams' button */
        createTeamsButton.addActionListener(e -> {
        createTeams(); /* createTeams method to generate teams */
             /* Create success label */
        JLabel successLabel = new JLabel("Teams have been created successfully!");
        successLabel.setBounds(50, 470, 300, 30); /* Set label position and size */
        mainFrame.add(successLabel); /* Add success label to mainFrame */
        mainFrame.repaint(); /* Repaint mainFrame to update GUI */
        });

        mainFrame.add(createTeamsButton); /* Add 'Create Teams' button to mainFrame */

        JButton weeklyGamesButton = new JButton("Weekly Matchups"); /* Create 'Weekly Matchups' button */
        weeklyGamesButton.setBounds(180, 500, 150, 30); /* Set button position and size */
        weeklyGamesButton.addActionListener(e -> {  /* Add action listener to 'Weekly Matchups' button */
            displayWeeklyMatchups(); /* displayWeeklyMatchups method to show matchups */
        });

        mainFrame.add(weeklyGamesButton); /* Add 'Weekly Matchups' button to mainFrame */

        JButton displayStatsButton = new JButton("Display Stats"); /* Create 'Display Stats' button */
        displayStatsButton.setBounds(350, 500, 150, 30); /* Set button position and size */
        displayStatsButton.addActionListener(e -> { /* Add action listener to 'Display Stats' button */
            displayStats(); /*  displayStats method to show match stats */
        });

        mainFrame.add(displayStatsButton); /* Add 'Display Stats' button to mainFrame */

        mainFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); /* Close operation for mainFrame */
        mainFrame.setVisible(true); /* Make mainFrame visible */
    }

    public static void createConsole(Team team) { /* Create a console window for the given team */
        JFrame consoleFrame = new JFrame(team.getName() + " Console");  /* Create a new JFrame with the team's name followed by "Console" */
        consoleFrame.setSize(600, 400); /* Set size of the console frame */
        consoleFrame.setLayout(null); /*  null layout for positioning */

        JTextArea textArea = new JTextArea(); /* Create a JTextArea to display player information */
        textArea.setBounds(10, 10, 560, 300); /* Set bounds for the text area */
        for (Player player : team.getPlayers()) { /* Loop through the players in the team and their information to the text area */
            textArea.append(player.toString() + "\n"); /* player info then comes a newline */
        }

        JButton returnButton = new JButton("Return");  /* Create a 'Return' button */
        returnButton.setBounds(250, 320, 100, 30); /* Set bounds for the return button */
        returnButton.addActionListener(e -> {  /* Add action listener to the 'Return' button */
            consoleFrame.dispose();  /* Dispose of the console frame */
            mainFrame.setVisible(true); /* Now make the main frame visible */
        });

        consoleFrame.add(textArea); /* Add the text area to the console frame */
        consoleFrame.add(returnButton); /* Add the return button to the console frame */
        consoleFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); /* Close the  operation for the console frame */
        consoleFrame.setVisible(true); /* Make the console frame visible */
    }

    public static void readPlayersFromFile(String filename) { /* Read players from the specified file */
        try (Scanner scanner = new Scanner(new File(filename))) {
            while (scanner.hasNextLine()) { /* Loop through each line in the file */
                String line = scanner.nextLine(); /* Read the next line */
                String[] parts = line.split(", "); /* Split the line into parts based on ", " */
                if (parts.length == 4) { /* Check if the line has exactly 4 parts */
                    String name = parts[0]; /* Get the name */
                    String position = parts[1]; /* Get the position */
                    int age = Integer.parseInt(parts[2]); /* Parse the age */
                    int number = Integer.parseInt(parts[3]);  /* Parse the number */

                    Player player = new Player(name, number, age, position); /* Create a new player with the extracted information */

                    /* Add the player accordingly based on their position */
                    switch (position) {
                        case "Goalkeeper":
                            goalkeepers.add(player); /* Add to goalkeepers list */
                            break;
                        case "Defender":
                            defenders.add(player); /* Add to defenders list */
                            break;
                        case "Midfielder":
                            midfielders.add(player); /* Add to midfielders list */
                            break;
                        case "Striker":
                            strikers.add(player); /* Add to strikers list */
                            break;
                        default:   /* Print an error message for invalid position */
                            System.out.println("Invalid position: " + position + " in line: " + line);
                            break;
                    }
                } else { /* Print an error message for invalid line format */
                    System.out.println("Invalid line format: " + line);
                }
            }
        } catch (Exception e) { /* Trying to catch an error */
            System.out.println("Error reading players from file: " + e.getMessage()); /* Print an error message if an exception occurs while rea
        }
    }

    public static void createTeams() { /* Create teams from the available players */
        if (goalkeepers.size() < 20 || defenders.size() < 80 || midfielders.size() < 80 || strikers.size() < 40) { /* Check if there are enough players to create teams */
            System.out.println("Not enough players to create teams.");
            return; /* Exit the method if not enough players */
        }

        Random random = new Random(); /* Created a Random object for selecting players randomly */
        teams.clear(); /* Clear the existing list of teams */

              /* Loop to create 20 teams */
        for (int i = 0; i < 20; i++) {
            Team team = new Team(teamNames[i]); /* Create a new team with a name from teamNames */
            team.addPlayer(goalkeepers.remove(random.nextInt(goalkeepers.size())));  /* Add a randomly selected goalkeeper to the team */
            for (int j = 0; j < 4; j++) { /* Loop to add 4 defenders and 4 midfielders to the team */
                team.addPlayer(defenders.remove(random.nextInt(defenders.size())));   /* Add a randomly selected defender */
                team.addPlayer(midfielders.remove(random.nextInt(midfielders.size()))); /* Add a randomly selected midfielder */
            }
            for (int j = 0; j < 2; j++) {  /* Loop to add 2 strikers to the team */
                team.addPlayer(strikers.remove(random.nextInt(strikers.size()))); /* Randomly add 2 strikers */
            }
            teams.add(team); /* Add the created team to the list of teams */
        }
    }

        /* Generate all possible matchups between teams */
    public static void generateAllMatchups() {
        allMatchups.clear(); /* Clear the existing list of matchups */
        for (int i = 0; i < teamNames.length; i++) {  /* Loop to generate matchups */
            for (int j = i + 1; j < teamNames.length; j++) {
                allMatchups.add(new String[]{teamNames[i], teamNames[j]}); /* Add a matchup between team i and team j */
            }
        }

          /* Shuffling the list for matchups */
        Random random = new Random(); /* Create a Random object for shuffling */
        for (int i = allMatchups.size() - 1; i > 0; i--) {
            int index = random.nextInt(i + 1); /* Select a random index */
            String[] temp = allMatchups.get(index); /* Swap them */
            allMatchups.set(index, allMatchups.get(i));
            allMatchups.set(i, temp);
        }
    }

        /* Display the matchups for the current week */
    public static void displayWeeklyMatchups() {
        if (teams.size() < 2) {  /* Check if there are enough teams to create matchups */
            JOptionPane.showMessageDialog(mainFrame, "Not enough teams to create matchups.");
            return; /* Exit the method if not enough teams */
        }

        JFrame matchupsFrame = new JFrame("Weekly Matchups"); /* Create a new frame to display the weekly matchups */
        matchupsFrame.setSize(400, 600);
        matchupsFrame.setLayout(null);

        JTextArea textArea = new JTextArea(); /* Create a text area to display the matchups */
        textArea.setBounds(10, 10, 360, 500);
        textArea.append("Weekly Matchups:\n");

        currentWeeklyGames.clear();  /* Clear previous week's games */

        /* ArrayList to keep track of teams that have been matched */
        ArrayList<String> matchedTeams = new ArrayList<>();
        int matchupsCount = 0; /* Counter for the number of matchups created */

         /* Loop to create 10 matchups */
        while (matchupsCount < 10) {
            if (currentMatchupIndex >= allMatchups.size()) {  /*  Shuffle the matchups if all have been used */
                currentMatchupIndex = 0; /* Reset index */
                shuffleArrayList(allMatchups); /* Shuffle to generate new matchups than last time */
            }
               /* Get the next matchup */
            String[] matchup = allMatchups.get(currentMatchupIndex++);
            String team1 = matchup[0];
            String team2 = matchup[1];
                 /* Checks if both teams have not been matched yet */
            if (!matchedTeams.contains(team1) && !matchedTeams.contains(team2)) {
                textArea.append(team1 + " vs " + team2 + "\n"); /* Display the matchup */
                currentWeeklyGames.add(game.Game(findTeamByName(team1), findTeamByName(team2))); /* Add the matchup to the current weekly games */
                matchedTeams.add(team1); /* Mark team1 as matched */
                matchedTeams.add(team2); /* Mark team2 as matched */
                matchupsCount++; /* Increment the matchups count */
            }
        }

                /* Add the text area to the frame and display the frame */
        JScrollPane scrollPane = new JScrollPane(textArea);
        scrollPane.setBounds(10, 10, 360, 500);
        matchupsFrame.add(scrollPane);

        JButton returnButton = new JButton("Return"); /* Creates a Return button */
        returnButton.setBounds(150, 530, 100, 30);
        returnButton.addActionListener(e -> {
            matchupsFrame.dispose();
            mainFrame.setVisible(true);
        });

        matchupsFrame.add(returnButton);
        matchupsFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        matchupsFrame.setVisible(true);
    }

    public static void displayStats() { /* Display the statistics for the current week's matchups */
        if (currentWeeklyGames.isEmpty()) { /* Check if there are any matchups to display stats for */
            JOptionPane.showMessageDialog(mainFrame, "No matchups to display stats for.");
            return; /* Exit the method if no matchups */
        }

             /* Create a new frame to display the match stats */
        JFrame statsFrame = new JFrame("Match Stats");
        statsFrame.setSize(400, 600);
        statsFrame.setLayout(null);

         /* Create a text area to display the stats */
        JTextArea textArea = new JTextArea();
        textArea.setBounds(10, 10, 360, 500);
        textArea.append("Match Stats:\n");

        Random random = new Random(); /* Random object to generate random scores */

             /* Loop through each game in the current weekly games */
        for (Game game : currentWeeklyGames) {
            int score1 = random.nextInt(5); /* Generate a random score for the first team */
            int score2 = random.nextInt(5); /* Generate a random score for the second team */
            /* Add the match results to the text area */
            textArea.append(game.getTeam1().getName() + " " + score1 + " - " +
                            score2 + " " + game.getTeam2().getName() + "\n");

              /* Add the text area to the frame and display the frame */
            if (score1 > score2) {
                game.getTeam1().addWin(); /* Add a win to the first team */
            } else if (score2 > score1) {
                game.getTeam2().addWin(); /* Check if the second team's score is greater than the first team's score */
            }


            game.getTeam1().addGoalsScored(score1); /* Add the goals scored by the first team to their total goals */
            game.getTeam2().addGoalsScored(score2); /* Add the goals scored by the second team to their total goals */

            if (score2 == 0) { /* Check if the second team scored zero goals */
                game.getTeam1().addCleanSheet(); /* Add a clean sheet to the first team */
            }
            if (score1 == 0) { /* Check if the first team scored zero goals */
                game.getTeam2().addCleanSheet(); /* Add a clean sheet to the second team */
            }
        }

        JScrollPane scrollPane = new JScrollPane(textArea); /* Create a scroll pane to hold the text area */
        scrollPane.setBounds(10, 10, 360, 500); /* Set the bounds of the scroll pane */
        statsFrame.add(scrollPane); /* Add the scroll pane to the stats frame */

        JButton rankByWinsButton = new JButton("Rank by Wins"); /* Create a button to rank teams by wins */
        rankByWinsButton.setBounds(10, 530, 150, 30); /* Set the bounds of the rank by wins button */
        rankByWinsButton.addActionListener(e -> { /* Add an action listener to the rank by wins button */
            rankTeamsByWins(); /* Rank the teams by their wins when the button is clicked */
        });

        statsFrame.add(rankByWinsButton); /* Add the rank by wins button to the stats frame */

        JButton rankByCleanSheetsButton = new JButton("Rank by Clean Sheets"); /* Create a button to rank teams by clean sheets */
        rankByCleanSheetsButton.setBounds(170, 530, 200, 30); /* Set the bounds of the rank by clean sheets button */
        rankByCleanSheetsButton.addActionListener(e -> { /* Add an action listener to the rank by clean sheets button */
            rankTeamsByCleanSheets(); /* Rank the teams by their clean sheets when the button is clicked */
        });

        statsFrame.add(rankByCleanSheetsButton); /* Add the rank by clean sheets button to the stats frame */

        JButton rankByGoalsButton = new JButton("Rank by Goals"); /* Create a button to rank teams by goals */
        rankByGoalsButton.setBounds(10, 570, 150, 30); /* Set the bounds for the rank by goals button */
        rankByGoalsButton.addActionListener(e -> { /* Add an action listener to the rank by goals button */
            rankTeamsByGoals(); /* Rank the teams by their goals when the button is clicked */
        });

        statsFrame.add(rankByGoalsButton); /* Add the rank by goals button to the stats frame */

        JButton returnButton = new JButton("Return"); /* Create a return button to go back to the main frame */
        returnButton.setBounds(170, 570, 100, 30); /* Set the bounds of the return button */
        returnButton.addActionListener(e -> { /* Add an action listener to the return button */
```

```java
            statsFrame.dispose(); /* Dispose of the stats frame and make the main frame visible again */
            mainFrame.setVisible(true);
        });

        statsFrame.add(returnButton); /* Add the return button to the stats frame */
        statsFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        statsFrame.setVisible(true);
    }

    public static void rankTeamsByWins() {
        JFrame rankingFrame = new JFrame("Team Rankings");
        rankingFrame.setSize(400, 600);
        rankingFrame.setLayout(null);

        JTextArea textArea = new JTextArea();
        textArea.setBounds(10, 10, 360, 500);
        textArea.append("Team Rankings by Wins:\n");

        ArrayList<Team> sortedTeams = new ArrayList<>(teams);
        sortedTeams.sort((t1, t2) -> t2.getWins() - t1.getWins());

        for (Team team : sortedTeams) {
            textArea.append(team.getName() + " - Wins: " + team.getWins() + "\n");
        }

        JScrollPane scrollPane = new JScrollPane(textArea);
        scrollPane.setBounds(10, 10, 360, 500);
        rankingFrame.add(scrollPane);

        JButton returnButton = new JButton("Return");
        returnButton.setBounds(150, 530, 100, 30);
        returnButton.addActionListener(e -> {
            rankingFrame.dispose();
            mainFrame.setVisible(true);
        });

        rankingFrame.add(returnButton);
        rankingFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        rankingFrame.setVisible(true);
    }

    /* Method to rank teams by their wins */
    public static void rankTeamsByCleanSheets() {
        JFrame rankingFrame = new JFrame("Team Rankings"); /* Create a new frame for displaying team rankings */
        rankingFrame.setSize(400, 600);
        rankingFrame.setLayout(null);

        JTextArea textArea = new JTextArea();  /* Create a text area to display the rankings */
        textArea.setBounds(10, 10, 360, 500);
        textArea.append("Team Rankings by Clean Sheets:\n");  /* Add a header for the rankings */

        ArrayList<Team> sortedTeams = new ArrayList<>(teams); /* Create a list of teams and sort them most wins to least wins */
        sortedTeams.sort((t1, t2) -> t2.getCleanSheets() - t1.getCleanSheets());

        for (Team team : sortedTeams) {  /* Append each team's name and win count to the text area */
            textArea.append(team.getName() + " - Clean Sheets: " + team.getCleanSheets() + "\n");
        }

        JScrollPane scrollPane = new JScrollPane(textArea); /* Create a scroll pane for the text area */
        scrollPane.setBounds(10, 10, 360, 500);
        rankingFrame.add(scrollPane);

        JButton returnButton = new JButton("Return"); /* Create a return button to go back to the main frame */
        returnButton.setBounds(150, 530, 100, 30);
        returnButton.addActionListener(e -> {
            rankingFrame.dispose();  /* Dispose of the ranking frame and make the main frame visible again */
            mainFrame.setVisible(true);
        });

        rankingFrame.add(returnButton); /* Add the return button to the ranking frame */
        rankingFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);   /* Set the default close operation of the ranking frame */
        rankingFrame.setVisible(true); /* Make the ranking frame visible */
    }

    public static void rankTeamsByGoals() { /* Method to rank teams by goals scored */
        JFrame rankingFrame = new JFrame("Team Rankings");   /* Create a new frame for displaying team rankings */
        rankingFrame.setSize(400, 600);
        rankingFrame.setLayout(null);

        JTextArea textArea = new JTextArea();  /* Create a text area to display the rankings */
        textArea.setBounds(10, 10, 360, 500);
        textArea.append("Team Rankings by Goals:\n");

        ArrayList<Team> sortedTeams = new ArrayList<>(teams);   /* Create a list of teams and sort them by most goals to least goals */
        sortedTeams.sort((t1, t2) -> t2.getGoalsScored() - t1.getGoalsScored());

        /* Add each team's name and goals scored count to the text area */
        for (Team team : sortedTeams) {
            textArea.append(team.getName() + " - Goals: " + team.getGoalsScored() + "\n");
        }

        JScrollPane scrollPane = new JScrollPane(textArea);   /* Create a scroll pane for the text area */
        scrollPane.setBounds(10, 10, 360, 500);
        rankingFrame.add(scrollPane);

        JButton returnButton = new JButton("Return"); /* Create a return button to go back to the main frame */
        returnButton.setBounds(150, 530, 100, 30);
        returnButton.addActionListener(e -> {
            rankingFrame.dispose();  /* Dispose of the ranking frame and make the main frame visible again */
            mainFrame.setVisible(true);
        });

        rankingFrame.add(returnButton);  /* Add the return button to the ranking frame */
        rankingFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);   /* Set the default close operation of the ranking frame */
        rankingFrame.setVisible(true); /* Make the ranking frame visible */
    }

    public static Team findTeamByName(String name) { /* Method to find a team by its name */
        for (Team team : teams) { /* Read through the list of teams */
            if (team.getName().equals(name)) { /* Check if the current team's name matches the given name */
                return team;
            }
        }
        return null; /* Return null if no matching team is found */
    }

    public static void shuffleArrayList(ArrayList<String[]> list) { /* Method to shuffle an ArrayList of string arrays */
        Random random = new Random();  /* Create a new Random object for generating random numbers */
        for (int i = list.size() - 1; i > 0; i--) { /* Loop through the list from the end to the beginning */
            int index = random.nextInt(i + 1);  /* Generate a random index */
            String[] temp = list.get(index);  /* Swap the current index for a random index */
            list.set(index, list.get(i));
            list.set(i, temp);
        }
    }

    public static ArrayList<Game> MakeGames(ArrayList<Team> teams) { /* Method to create a list of games from a list of teams */
        ArrayList<Game> games = new ArrayList<>(); /* Create a new ArrayList to store the games */
        ArrayList<Team> team = new ArrayList<>(teams);  /* Create a copy of the inputed teams list */
        int value;
        Random rand = new Random();
        while (team.size() != 0) {  /* Loop until there are no more teams left in the list */
            value = rand.nextInt(team.size()); /* Generate a random index and select a team */
            Team team1 = team.get(value);
            team.remove(value);  /* Remove the selected team from the list */
            value = rand.nextInt(team.size());  /* Generate another random index and select another team */
            Team team2 = team.get(value);
            team.remove(value);       /* Remove the second selected team from the list */
            Game gamePlayed =  game.Game(team1, team2);  /* Create a new game with the two selected teams */
            games.add(gamePlayed);   /* Add the created game to the list of games */
        }
        return games; /* Return the list of created games */
    }
}
```

Game.java

```java
class Game {

    Team team1;
    Team team2;

    /* Constructor to initialize a new game with the specified teams */
    public Game Game(Team team1, Team team2) {
        Game game = new Game();
        game.team1 = team1;
        game.team2 = team2;
        return game;
    } /* End of Game constructor */

    /* Method to get the first team in the game */
    public Team getTeam1() {
        return team1;
    } /* End of getTeam1 method */

    /* Method to get the second team in the game */
    public Team getTeam2() {
        return team2;
    } /* End of getTeam2 method */

    /* Method to return a string representation of the game */
    public String toString() {

        return team1.getName() + " vs " + team2.getName();
    } /* End of toString method */
} /* End of Game class */
```

Main.java

```java
import java.util.Scanner;
import java.util.ArrayList;

public class Main {
    /* Main method to execute the program */
    public static void main(String[] args){
        System.out.print(makeImagesList()); /* Print the list of image filenames */
    } /* End of main method */

    /* Method to create a list of image filenames from a file */
    public static ArrayList<String> makeImagesList(){

        /* Create a scanner to read from the "image.txt" file */
        Scanner scan = new Scanner(Main.class.getResourceAsStream("image.txt"));
        scan.useDelimiter(","); /* Set the delimiter to comma */

        /* Initialize the list to hold image filenames */
        ArrayList<String> images = new ArrayList<String>();

        /* Read each line from the file and add to the list */
        while(scan.hasNextLine())
            images.add(scan.next()); /* Add the next token to the list */

        return images; /* Return the list of image filenames */

    } /* End of makeImagesList method */
} /* End of Main class */
```

Manager.java

```
import java.util.ArrayList;

/* Class representing a manager for handling games */
public class Manager {

   /* List to hold the games */
   public ArrayList<Game> games;

} /* End of Manager class */
```

Manager.txt

```
public class Manager {
    public static void main(String[] args) {
    }
}
```

Player.java

```
class Player {

    public String name;
    public int number;
    public int age;
    public String position;

    /* Specify and initialize a new player with the specified attributes */
    public Player(String name, int number, int age, String position) {
        this.name = name;
        this.number = number;
        this.age = age;
        this.position = position;
    } /* End of Player constructor */

    /* Method to get the player's name */
    public String getName() {
        return name;
    } /* End of getName method */

    /* Method to get the player's number */
    public int getNumber() {
        return number;
    } /* End of getNumber method */

    /* Method to get the player's age */
    public int getAge() {
        return age;
    } /* End of getAge method */

    /* Method to get the player's position */
    public String getPosition() {
        return position;
    } /* End of getPosition method */

    /* Method to return a string representation of the player */

    public String toString() {
        return name + " #" + number + " (" + position + ", " + age + " years old)";

    } /* End of toString method */
} /* End of Player class */
```

Player.txt

```
Prabjot Deol, Goalkeeper, 25, 1
Aurelia Rossi, Goalkeeper, 28, 13
Dimitri Petrov, Goalkeeper, 27, 22
Elena Vargas, Goalkeeper, 24, 7
Hugo Andersson, Goalkeeper, 26, 33
Nadia Kovalenko, Goalkeeper, 23, 10
Matteo Bianchi, Goalkeeper, 29, 1
Zara Al-Farsi, Goalkeeper, 22, 11
```

```
Lucas Costa, Goalkeeper, 25, 12
Anya Volkova, Goalkeeper, 21, 30
Rafael Morales, Goalkeeper, 30, 1
Isaac Okafor, Goalkeeper, 24, 24
Lena Schmidt, Goalkeeper, 27, 1
Kazuki Tanaka, Goalkeeper, 25, 7
Eva van der Berg, Goalkeeper, 26, 5
Nikolai Ivanov, Goalkeeper, 28, 99
Sofia Fernandez, Goalkeeper, 23, 8
Andrei Popescu, Goalkeeper, 31, 1
Freja Olsen, Goalkeeper, 22, 18
Diego Herrera, Goalkeeper, 29, 17
Jaskaran Singh, Defender, 24, 3
Luca Moretti, Defender, 22, 14
Emilio García, Defender, 27, 4
Nico Fischer, Defender, 25, 5
Julian Reyes, Defender, 23, 6
Sam O'Neill, Defender, 26, 2
Alexander Kim, Defender, 28, 15
Ismael Silva, Defender, 21, 16
Felix Martinez, Defender, 25, 17
Marco López, Defender, 24, 18
Oscar Johansson, Defender, 22, 19
Terry Wilson, Defender, 29, 20
Victor Hugo, Defender, 26, 21
Sergio Rossi, Defender, 27, 22
Tommy Nguyen, Defender, 24, 23
Lars Eriksen, Defender, 23, 24
Ethan Baker, Defender, 28, 25
Aiden Dubois, Defender, 25, 26
Carlos Pereira, Defender, 21, 27
Hans Müller, Defender, 26, 28
Louis Bertrand, Defender, 22, 29
Emil Andersson, Defender, 27, 30
Jacob Turner, Defender, 24, 31
Igor Petrov, Defender, 23, 32
Mateo Ricci, Defender, 25, 33
Lucas Santos, Defender, 22, 34
Dmitri Ivanov, Defender, 29, 35
Soren Kim, Defender, 26, 36
Evan Carter, Defender, 24, 37
Milo Novak, Defender, 23, 38
Julian Müller, Defender, 27, 39
Oliver Johnson, Defender, 21, 40
Leo Moretti, Defender, 28, 41
Ian Becker, Defender, 25, 42
Maxim Popov, Defender, 22, 43
Leon Fischer, Defender, 26, 44
Daniel Huang, Defender, 23, 45
Alex Ruiz, Defender, 24, 46
Sebastian Schwarz, Defender, 29, 47
Nate Costa, Defender, 21, 48
Andre Santos, Defender, 27, 49
Ethan Clark, Defender, 25, 50
Liam Walker, Defender, 22, 51
Ivan Ivanov, Defender, 23, 52
Jack Wilson, Defender, 26, 53
Anton Meier, Defender, 24, 54
Gabriel Oliveira, Defender, 28, 55
Chris Martin, Defender, 25, 56
Noah Kim, Defender, 21, 57
Valerio Bianchi, Defender, 26, 58
David Schneider, Defender, 23, 59
Leo Mendes, Defender, 24, 60
Miguel Fernandez, Defender, 29, 61
Sam White, Defender, 22, 62
Eli Cohen, Defender, 27, 63
Nathan Wright, Defender, 25, 64
Omar Hassan, Defender, 21, 65
Giulio Rossi, Defender, 23, 66
Tobias Wagner, Defender, 26, 67
James Khan, Defender, 24, 68
Hector Garcia, Defender, 28, 69
Ray Thompson, Defender, 25, 70
Aaron Taylor, Defender, 22, 71
Alec Ivanov, Defender, 23, 72
Ryan Lewis, Defender, 27, 73
Eric Smith, Defender, 24, 74
Christian Müller, Defender, 29, 75
Simon Nguyen, Defender, 21, 76
Leonardo Pereira, Defender, 26, 77
Mario Johansson, Defender, 25, 78
Hugo Becker, Defender, 23, 79
Nick Kim, Defender, 24, 80
Oscar Schmidt, Defender, 28, 81
Daniel Romero, Defender, 25, 82
Jasper Wang, Defender, 30, 83
Antonio Silva, Defender, 28, 84
Lucas Martin, Defender, 24, 85
Stefan Petrova, Defender, 29, 86
Nikolai Popov, Defender, 26, 87
Liam O'Connor, Defender, 27, 88
Ethan Reed, Midfielder, 24, 3
Caleb Jackson, Midfielder, 22, 14
Liam Thompson, Midfielder, 27, 4
Logan Harris, Midfielder, 25, 5
Owen Campbell, Midfielder, 23, 6
Lucas Martinez, Midfielder, 26, 2
```

```
107. Mason Walker, Midfielder, 29, 15
108. Oliver White, Midfielder, 21, 16
109. Elijah Rodriguez, Midfielder, 26, 17
110. Daniel Young, Midfielder, 25, 18
111. Michael King, Midfielder, 22, 19
112. William Adams, Midfielder, 29, 20
113. James Wright, Midfielder, 26, 21
114. Benjamin Hill, Midfielder, 27, 22
115. Alexander Turner, Midfielder, 24, 23
116. Matthew Scott, Midfielder, 21, 24
117. David Green, Midfielder, 28, 25
118. Joseph Baker, Midfielder, 25, 26
119. John Lewis, Midfielder, 21, 27
120. Nicholas Hall, Midfielder, 26, 28
121. Andrew Allen, Midfielder, 22, 29
122. Samuel Nelson, Midfielder, 27, 30
123. Christopher Carter, Midfielder, 24, 31
124. Jonathan Ward, Midfielder, 21, 32
125. Ryan Morris, Midfielder, 29, 33
126. Brandon Mitchell, Midfielder, 26, 34
127. Tyler Perez, Midfielder, 23, 35
128. Christian Roberts, Midfielder, 28, 36
129. Zachary Edwards, Midfielder, 25, 37
130. Nathan Collins, Midfielder, 21, 38
131. Justin Rivera, Midfielder, 27, 39
132. Gabriel Brooks, Midfielder, 24, 40
133. Kevin Hughes, Midfielder, 23, 41
134. Cameron Flores, Midfielder, 28, 42
135. Thomas Russell, Midfielder, 25, 43
136. Dylan James, Midfielder, 22, 44
137. Isaac Coleman, Midfielder, 27, 45
138. Aaron Powell, Midfielder, 26, 46
139. Jordan Jenkins, Midfielder, 21, 47
140. Kyle Gray, Midfielder, 26, 48
141. Jason Ramirez, Midfielder, 25, 49
142. Evan Watson, Midfielder, 22, 50
143. Austin Foster, Midfielder, 27, 51
144. Adam Gonzalez, Midfielder, 24, 52
145. Ian Butler, Midfielder, 23, 53
146. Cole Henderson, Midfielder, 28, 54
147. Adrian Ross, Midfielder, 25, 55
148. Julian Cooper, Midfielder, 21, 56
149. Jose Bailey, Midfielder, 26, 57
150. Nathaniel Morgan, Midfielder, 28, 58
151. Victor Reed, Midfielder, 23, 59
152. Alexandre Diaz, Midfielder, 24, 60
153. Dominic Murphy, Midfielder, 29, 61
154. Peter Foster, Midfielder, 25, 62
155. Juan Bryant, Midfielder, 22, 63
156. Levi Coleman, Midfielder, 27, 64
157. Hunter Simmons, Midfielder, 24, 65
158. Bryan Wood, Midfielder, 23, 66
159. Xavier Cox, Midfielder, 28, 67
160. Jackson Fisher, Midfielder, 25, 68
161. Colton Roberts, Midfielder, 22, 69
162. Carter Henderson, Midfielder, 27, 70
163. Sebastian Foster, Midfielder, 26, 71
164. Gavin Perry, Midfielder, 21, 72
165. Lucas Diaz, Midfielder, 23, 73
166. Jaden Barnes, Midfielder, 24, 74
167. Chase Marshall, Midfielder, 29, 75
168. Riley Gonzalez, Midfielder, 25, 76
169. Connor Brooks, Midfielder, 21, 77
170. Eli Patterson, Midfielder, 26, 78
171. Derek Ward, Midfielder, 25, 79
172. Brayden Ellis, Midfielder, 24, 80
173. Carter Hayes, Midfielder, 28, 81
174. Miles Jenkins, Midfielder, 22, 82
175. Hayden Hughes, Midfielder, 27, 83
176. Jack Ross, Midfielder, 25, 84
177. Maximilian Bryant, Midfielder, 21, 85
178. Julian Mitchell, Midfielder, 28, 86
179. Henry Bailey, Midfielder, 23, 87
180. Dominic Nelson, Midfielder, 24, 88
181. Om Patel, Striker, 24, 23
182. Luca Moretti, Striker, 22, 14
183. Emilio García, Striker, 27, 4
184. Nico Fischer, Striker, 25, 5
185. Julian Reyes, Striker, 23, 6
186. Sam O'Neill, Striker, 26, 2
187. Alexander Kim, Striker, 28, 15
188. Ismael Silva, Striker, 21, 16
189. Felix Martinez, Striker, 25, 17
190. Marco López, Striker, 24, 18
191. Oscar Johansson, Striker, 22, 19
192. Terry Wilson, Striker, 29, 20
193. Victor Hugo, Striker, 26, 21
194. Sergio Rossi, Striker, 27, 22
195. Tommy Nguyen, Striker, 24, 3
196. Lars Eriksen, Striker, 23, 24
197. Ethan Baker, Striker, 28, 25
198. Aiden Dubois, Striker, 25, 26
199. Carlos Pereira, Striker, 21, 27
200. Hans Müller, Striker, 26, 28
201. Louis Bertrand, Striker, 22, 29
202. Emil Andersson, Striker, 27, 30
203. Jacob Turner, Striker, 24, 31
204. Igor Petrov, Striker, 23, 32
```

```
Mateo Ricci, Striker, 25, 33
Lucas Santos, Striker, 22, 34
Dmitri Ivanov, Striker, 29, 35
Soren Kim, Striker, 26, 36
Evan Carter, Striker, 24, 37
Milo Novak, Striker, 23, 38
Julian Müller, Striker, 27, 39
Oliver Johnson, Striker, 21, 40
Leo Moretti, Striker, 28, 41
Ian Becker, Striker, 25, 42
Maxim Popov, Striker, 22, 43
Leon Fischer, Striker, 26, 44
Daniel Huang, Striker, 23, 45
Alex Ruiz, Striker, 24, 46
Sebastian Schwarz, Striker, 29, 47
Wyatt Kim, Striker, 21, 48
```

Team.java

```java
import java.util.ArrayList;

/* Class representing a team */
class Team {
    public String name;
    public ArrayList<Player> players;
    public int wins;
    public int cleanSheets;
    public int goalsScored;

    /* Constructor to initialize a new team with the specified name */
    public Team(String name) {
        this.name = name;
        this.players = new ArrayList<>();
        this.wins = 0;
        this.cleanSheets = 0;
        this.goalsScored = 0;
    } /* End of Team constructor */

    /* Method to get the team's name */
    public String getName() {
        return name;
    } /* End of getName method */

    /* Method to get the list of players in the team */
    public ArrayList<Player> getPlayers() {
        return players;
    } /* End of getPlayers method */

    /* Method to get the number of wins the team has */
    public int getWins() {
        return wins;
    } /* End of getWins method */

    /* Method to get the number of clean sheets the team has */
    public int getCleanSheets() {
        return cleanSheets;
    } /* End of getCleanSheets method */

    /* Method to get the number of goals scored by the team */
    public int getGoalsScored() {
        return goalsScored;
    } /* End of getGoalsScored method */

    /* Method to add a player to the team */
    public void addPlayer(Player player) {
        players.add(player);
    } /* End of addPlayer method */

    /* Method to add a win to the team's record */
    public void addWin() {
        wins++;
    } /* End of addWin method */

    /* Method to add a clean sheet to the team's record */
    public void addCleanSheet() {
        cleanSheets++;
    } /* End of addCleanSheet method */

    /* Method to add goals to the team's total goals scored */
    public void addGoalsScored(int goals) {
        goalsScored += goals;

    } /* End of addGoalsScored method */
} /* End of Team class */
```

image.txt

```
https://i.ibb.co/HgLGDPG/1.png
https://i.ibb.co/jJTRd2H/2.png
https://i.ibb.co/tZpvt9T/3.png
https://i.ibb.co/Gxgs4k3/4.png
https://i.ibb.co/ByKTMHy/5.png
https://i.ibb.co/3170h6P/6.png
https://i.ibb.co/2YddYV6/7.png
```

```
https://i.ibb.co/pxdQ1XB/215.png
https://i.ibb.co/zhZZrHL/216.png
https://i.ibb.co/hdsY7Zn/217.png
https://i.ibb.co/w6WcGK3/218.png
https://i.ibb.co/YLBzRby/219.png
https://i.ibb.co/khDFgrb/220.png
https://i.ibb.co/yFfV8cT/200.png
https://i.ibb.co/8Y7tjht/201.png
https://i.ibb.co/7rVKmSz/202.png
https://i.ibb.co/CVxBFpC/203.png
https://i.ibb.co/Pcch972/204.png
https://i.ibb.co/1KNfxjG/205.png
https://i.ibb.co/1GPTX42/206.png
https://i.ibb.co/2Z6Kn5N/207.png
https://i.ibb.co/RyNd08g/208.png
https://i.ibb.co/GWxLhtJ/209.png
https://i.ibb.co/Mn1DjCP/210.png
https://i.ibb.co/8MDNVtX/211.png
https://i.ibb.co/yYtXFWB/212.png
https://i.ibb.co/SvtBxdX/213.png
https://i.ibb.co/QMHZDRN/214.png
https://i.ibb.co/jM8Jmfp/185.png
https://i.ibb.co/s2K2ggy/186.png
https://i.ibb.co/dPTwPLG/187.png
https://i.ibb.co/XX3VQSy/188.png
https://i.ibb.co/wh8pT2d/189.png
https://i.ibb.co/48wC9J1/190.png
https://i.ibb.co/4wkKXHB/191.png
https://i.ibb.co/WW0NwFs/192.png
https://i.ibb.co/nkwGsyP/193.png
https://i.ibb.co/XJRFqsc/194.png
https://i.ibb.co/ZJ3k3JR/195.png
https://i.ibb.co/RYmqdYz/196.png
https://i.ibb.co/VwBWRzm/197.png
https://i.ibb.co/gt5W5nn/198.png
https://i.ibb.co/BzQgPNN/199.png
https://i.ibb.co/8jZ02Km/170.png
https://i.ibb.co/h8GfHvQ/171.png
https://i.ibb.co/rmXb8Jb/172.png
https://i.ibb.co/V3XTqXQ/173.png
https://i.ibb.co/BVYGMCm/174.png
https://i.ibb.co/cYZNdzR/175.png
https://i.ibb.co/MNdHz6n/176.png
https://i.ibb.co/kyfs78X/177.png
https://i.ibb.co/8BZs7DJ/178.png
https://i.ibb.co/v3tvfpQ/179.png
https://i.ibb.co/b1Zc545/180.png
https://i.ibb.co/KmpthtW/181.png
https://i.ibb.co/C2rrm6Q/182.png
https://i.ibb.co/mXqBBvN/183.png
https://i.ibb.co/r3NFdrC/184.png
https://i.ibb.co/b1qCCdG/155.png
https://i.ibb.co/3y0NQ4f/156.png
https://i.ibb.co/SRyBGzG/157.png
https://i.ibb.co/CtCWXQF/158.png
https://i.ibb.co/HzKPFv1/159.png
https://i.ibb.co/8NH0yqv/160.png
https://i.ibb.co/MgkZXPt/161.png
https://i.ibb.co/3TsyVpC/162.png
https://i.ibb.co/d258HX4/163.png
https://i.ibb.co/bRmPPYq/164.png
https://i.ibb.co/wNvB9Dc/165.png
https://i.ibb.co/F08pbcz/166.png
https://i.ibb.co/VY6Q3Xv/167.png
https://i.ibb.co/mcQH41c/168.png
https://i.ibb.co/hCfSKjB/169.png
https://i.ibb.co/qNXf7ZF/140.png
https://i.ibb.co/q02s718/141.png
https://i.ibb.co/2898gFv/142.png
https://i.ibb.co/mNGLQ3Q/143.png
https://i.ibb.co/pL9J85y/144.png
https://i.ibb.co/gmB6j1Q/145.png
https://i.ibb.co/zhQCz9J/146.png
https://i.ibb.co/VWZj9wW/147.png
https://i.ibb.co/10x7BVg/148.png
https://i.ibb.co/vzMj2SS/149.png
https://i.ibb.co/C2M6Mmh/150.png
https://i.ibb.co/zXH13dg/151.png
https://i.ibb.co/KLSq248/152.png
https://i.ibb.co/7N0Rxzz/153.png
https://i.ibb.co/PgqK5fp/154.png
https://i.ibb.co/ygXvhj4/125.png
https://i.ibb.co/tLHs3MV/126.png
https://i.ibb.co/XC6vrvh/127.png
https://i.ibb.co/9G2612x/128.png
https://i.ibb.co/GRLsgnt/129.png
https://i.ibb.co/CV0WDvS/130.png
https://i.ibb.co/K0R6qHY/131.png
https://i.ibb.co/XCBmjrF/132.png
https://i.ibb.co/dr2MGhx/133.png
https://i.ibb.co/JCMx7Bd/134.png
https://i.ibb.co/R2Vh1ZM/135.png
https://i.ibb.co/1dnYQcp/136.png
https://i.ibb.co/GTx23Wk/137.png
https://i.ibb.co/3zhkxKq/138.png
https://i.ibb.co/jvRvcv1/139.png
https://i.ibb.co/4N4qqY0/110.png
https://i.ibb.co/WsqgsYx/111.png
```

```
https://i.ibb.co/K6jSTXC/112.png
https://i.ibb.co/0DgZZxj/113.png
https://i.ibb.co/MPj3YQv/114.png
https://i.ibb.co/3WsHZFj/115.png
https://i.ibb.co/WyH6Lc6/116.png
https://i.ibb.co/7gZdKbQ/117.png
https://i.ibb.co/yh7bmGj/118.png
https://i.ibb.co/xCymzk6/119.png
https://i.ibb.co/0VCSBMB/120.png
https://i.ibb.co/KxR59v7/121.png
https://i.ibb.co/SQ4PbFY/122.png
https://i.ibb.co/tz09HNT/123.png
https://i.ibb.co/VqJ03hB/124.png
https://i.ibb.co/sFQvNvB/95.png
https://i.ibb.co/mR8SsrV/96.png
https://i.ibb.co/tHbk2gb/97.png
https://i.ibb.co/bKjs5s5/98.png
https://i.ibb.co/tLC2S0B/99.png
https://i.ibb.co/TLXyJvn/100.png
https://i.ibb.co/ZMrKtcp/101.png
https://i.ibb.co/1ZLNbJb/102.png
https://i.ibb.co/m0SwWBB/103.png
https://i.ibb.co/jRcqJHc/104.png
https://i.ibb.co/0DpCPpZ/105.png
https://i.ibb.co/vzvGJvG/106.png
https://i.ibb.co/yRdDdHy/107.png
https://i.ibb.co/Jcqc0LQ/108.png
https://i.ibb.co/sjb9vxn/109.png
https://i.ibb.co/hfBRc40/80.png
https://i.ibb.co/ZSZ8JBP/81.png
https://i.ibb.co/3c89bVs/82.png
https://i.ibb.co/SvJFtKh/83.png
https://i.ibb.co/9HRzGfS/84.png
https://i.ibb.co/1MF5xN0/85.png
https://i.ibb.co/DC9Djpr/86.png
https://i.ibb.co/JnWBLcq/87.png
https://i.ibb.co/LJh72gH/88.png
https://i.ibb.co/ZT2fwWN/89.png
https://i.ibb.co/LnMTSGn/90.png
https://i.ibb.co/wsDj9XK/91.png
https://i.ibb.co/RBz6nGs/92.png
https://i.ibb.co/N2qHvRS/93.png
https://i.ibb.co/3sXrW9H/94.png
https://i.ibb.co/tKPQY0N/65.png
https://i.ibb.co/RYQ0vWq/66.png
https://i.ibb.co/Xyjnh43/67.png
https://i.ibb.co/27TYNYj/68.png
https://i.ibb.co/NNwzg7X/69.png
https://i.ibb.co/F7RpdBK/70.png
https://i.ibb.co/dtTQQ2g/71.png
https://i.ibb.co/djnyhVW/72.png
https://i.ibb.co/nzNZK9T/73.png
https://i.ibb.co/9srtNyH/74.png
https://i.ibb.co/bRQtPb9/75.png
https://i.ibb.co/b7kc7V5/76.png
https://i.ibb.co/wzZQb8k/77.png
https://i.ibb.co/cLJPsN3/78.png
https://i.ibb.co/zXRKvjg/79.png
https://i.ibb.co/2dhBFY9/50.png
https://i.ibb.co/tzZTxyq/51.png
https://i.ibb.co/ZMt85ys/52.png
https://i.ibb.co/T2vJbgZ/53.png
https://i.ibb.co/KNF71fD/54.png
https://i.ibb.co/9pgjBYG/55.png
https://i.ibb.co/bKRKBMk/56.png
https://i.ibb.co/sC2bcC7/57.png
https://i.ibb.co/Dgf6Zwy/58.png
https://i.ibb.co/NyRQRt8/59.png
https://i.ibb.co/3NkvNWf/60.png
https://i.ibb.co/yygPx29/61.png
https://i.ibb.co/r44Nbcm/62.png
https://i.ibb.co/3vdw5xN/63.png
https://i.ibb.co/kyr3gjn/64.png
https://i.ibb.co/yqd94tg/35.png
https://i.ibb.co/z86xD2s/36.png
https://i.ibb.co/nkRpSGt/37.png
https://i.ibb.co/sPv90YZ/38.png
https://i.ibb.co/pPPdDLX/39.png
https://i.ibb.co/Zzw9wv9/40.png
https://i.ibb.co/3sNgFXK/41.png
https://i.ibb.co/x8F56mR/42.png
https://i.ibb.co/tcnVVwv/43.png
https://i.ibb.co/qNVtJKd/44.png
https://i.ibb.co/sKD1jLh/45.png
https://i.ibb.co/kG2R9fc/46.png
https://i.ibb.co/Lk9jmhm/47.png
https://i.ibb.co/z8YjcZv/48.png
https://i.ibb.co/YWpJZWC/49.png
https://i.ibb.co/K2rSbz9/21.png
https://i.ibb.co/S6NfLvS/22.png
https://i.ibb.co/tmKZp98/23.png
https://i.ibb.co/mzRVQF6/24.png
https://i.ibb.co/m6RF10G/25.png
https://i.ibb.co/NS6Rjbs/26.png
https://i.ibb.co/LZHSh0p/27.png
https://i.ibb.co/2yPdVfS/28.png
https://i.ibb.co/r0ZvT0X/29.png
https://i.ibb.co/bgL297p/30.png
```

```
https://i.ibb.co/qmB8vjB/31.png
https://i.ibb.co/gtPqHdV/32.png
https://i.ibb.co/hMYTYPS/33.png
https://i.ibb.co/TmGm5LR/34.png
https://i.ibb.co/ZGGxw9h/8.png
https://i.ibb.co/t4JKNZd/9.png
https://i.ibb.co/6r7cgCL/10.png
https://i.ibb.co/TRs3GWd/11.png
https://i.ibb.co/R64CPT0/12.png
https://i.ibb.co/wQGF7xF/13.png
https://i.ibb.co/kmvQH0H/14.png
https://i.ibb.co/TgpR1g9/15.png
https://i.ibb.co/m98hxxv/16.png
https://i.ibb.co/zZb5XF6/17.png
https://i.ibb.co/G7xSjZm/18.png
https://i.ibb.co/550whPx/19.png
https://i.ibb.co/Qm9Xm7n/20.png
```