**Comparable – Player Ranking**

1. Write a program to display the ranking list of IPL players based on their scores ranked from highest to lowest.   
Comparable interface is used to define the natural ordering of objects. Implement Comparable interface and implement compareTo() method which returns a negative integer, zero, or a positive integer as this object is less than, equal to, or greater than the specified object.   
  
Get all the player names and their corresponding scores from the user and store it in a list. Use Collections.sort() method to sort the list and display the result as shown in the sample input and output.

Create **Ranking** class with the private attributes

name(String) and score(long/int) implementing Comparable interface and implementing compareTo() method

Include a constructor with the arguments player name and score   
Include appropriate getter and setter for the Ranking class   
  
Create a main class "**RankingMain.java**"   
  
  
  
**Input and Output Format:**   
First input corresponds to the number of players.   
Display the ranking details separated by single space.   
Refer sample input and output for formatting specifications.   
  
**[All text in bold corresponds to input and the rest corresponds to output]**   
**Sample Input/Output :**   
Please provide the number of  players   
**3**   
Enter the name of the player 1   
**Kohli**   
Enter the score of the player 1   
**1093**   
Enter the name of the player 2   
**Dhoni**   
Enter the score of the player 2   
**1135**   
Enter the name of the player 3   
**Ashwin**   
Enter the score of the player 3   
**335**   
Player Details by Score(High to Low)   
1 Dhoni 1135   
2 Kohli  1093   
3 Ashwin  335

**Comparable - List Match By Date**

2. Write a java program to get all the match details from the user and display the match details sorted based on date in descending order. The match details consist of match date,  teamOne and teamTwo.   
   
Create Match class with below members,

* matchDate – util.Date or LocalDate or String
* teamOne - String
* teamTwo - String

Add appropriate getter and setter methods for Match class   
*Example:*   
*Attribute - matchDate*   
*Method - getMatchDate (), setMatchDate (String matchDate)*   
Include  a constructor for Match with the arguments matchDate, teamOne, teamTwo   
Implement Comparable interface and implement compareTo method to do the sorting based on the date   
  
Create a main class "MatchSortMain.java"

**Input and Output Format:**   
First input corresponds to the number of matches and followed by each match information.   
Refer sample input and output for formatting specifications.   
  
**[All text in bold corresponds to input and the rest corresponds to output]**   
**Sample Input/Output :**   
Enter the number of matches   
**3**   
Enter  match date in (MM-dd-yyyy)   
**10-20-2020**   
Enter Team 1   
**RCB**   
Enter Team 2   
**KKR**   
Enter  match date in (MM-dd-yyyy)   
**09-23-2020**   
Enter Team 1   
**MI**   
Enter Team 2   
**KKR**   
Enter  match date in (MM-dd-yyyy)   
**10-10-2020**   
Enter Team 1   
**CSK**   
Enter Team 2   
**RCB**

Match Details

Team 1 RCB

Team 2 KKR

Match held on 10-20-2020

Team 1 CSK

Team 2 RCB

Match held on 10-10-2020

Team 1 MI

Team 2 KKR

Match held on 09-23-2020

**Comparable - Display Team**

Write a Java program to get the team names and player names from the user seperated by a pipe symbol. Finally display all the teams and their players sorted in ascending order based on their names.

Create **Player class** with single private attribute name

Add appropriate getter and setter methods for Player class

Include a constructor with single argument Player name

Implement Comparable interface in the Player class and implement the method compareTo()

Create **Team clas**s with below private attributes,

name - String

playerList - List<Player> (All player object for this team is stored in this list)

Add appropriate getter and setter methods for Team class

Include a constructor accepting Team name as a parameter

Below are the methods in Team class

addPlayer(String playername) - Add the new player to this team object

getPlayerList() - Sort the player collection and return the list

Create a main class "**DisplayTeamMain.java**"

**Input and Output Format:**

First input corresponds to the number of input elements and followed by team and player information in the format teamname|playername.

Display the player name followed by two hyphen(-)

Refer sample input and output for formatting specifications.

**Sample Input/Output :**

10

CSK|MS Dhoni

CSK|Sam Curran

CSK|Ambati Rayudu

RCB|Virat Kohli

RCB|Davudatt Padikkal

RCB|AB de Villiers

CSK|Suresh Raina

KKR|Eoin Morgan

KKR|Pat Cummins

KKR|Dinesh Karthick

Team and Players in ascending order

CSK

--Ambati Rayudu

--MS Dhoni

--Sam Curran

--Suresh Raina

KKR

--Dinesh Karthick

--Eoin Morgan

--Pat Cummins

RCB

--AB de villiers

--Devudatt Padikkal

--Virat Kohli