Today, you are going to create a NcaaTeam class that stores the following:

String name, ArrayList<NcaaTeam> schedule, ArrayList<Score> results, private double winRatio

//Schedule is an ArrayList that contains the ncaa teams in the order that they play them this season

//Results stores Score objects:

```
public class Score
{
        private NcaaTeam team1;
        private NcaaTeam team2;
        private int score1; //team1score
        private int score2; //team2score
public Score(NcaaTeam team1, NcaaTeam team2, int score1, int score2)
{
        this.team1 = team1;
        this.team2 = team2;
        this.score1 = score1;
        this.score2 = score2;
}
public NcaaTeam obtainTeam(int num){
        if(num==1) return team1;
        else return team2;
}
public NcaaTeam getScore(int num){
        if(num==1) return score1;
        else return score2;
}}
```

1. Create a method that constructs a NcaaTeam initially being sent the team name. It should instantiate appropriate objects and set the winRatio at 0%.
   For example:
   NcaaTeam gt = new NcaaTeam("Georgia Tech");

2. Create a method that allows one to add new games to the schedule in NcaaTeam for both teams.

   For example:
   gt.scheduleAdd(uga);
   //will add a game against uga to gt's schedule AND a game against gt to uga's schedule

3. Create a method that updates the results arraylist scores anytime new scoring criteria (teams and scores are sent. You can assume game results will be sent in order of schedule (so you do not need to update or match to the schedule array).

   For example:

   gt.addscore(uga, 71, 75)

//This will add a new score record to gt’s results in which gt scored 71 and uga scored 75. It will also add this score record reference (without instantiating it again) to uga’s results arraylist.

4. Create a method that when called computes a win percentage based on the games played thus far and updates the winRatio field. It will not include games that have not been played. It will return the win percentage.

   For example:

   double gtwin = gt.computeWin();

   //Please note, score records can store teams in any order, so you will need some conditionals to match teams and evaluate who is the winner. Assume ties are not allowed in games!

   Solution (I’ve added print statements so you can see how this works if you have struggling students):

   https://repl.it/@LauraGray/MarchMadness

Grading Rubric:

Point	Description:
Class Level Variables & Constructor – 4 pts	
+1	Student declares appropriate class level variables (1/2) and sets them to private (1/2).
+1	Student creates constructor method name correctly with correct parameters
+1	Student assigns correct parameters to class level variables in constructor, including setting the WinRatio to 0.0 (can be done at class level variable as well)
+1	Student correctly instantiates the schedule and results ArrayLists in constructor (Could be done at the class level variable, though they will not be able to do this on an AP exam where the variables are already declared).
addSchedule Method – 3 pts	
+1	Student correctly names method with correct parameters and objects returned
+1	Student adds NcaaTeam team parameter to the NcaaTeam that called the object’s schedule ArrayList
+1	Student adds NcaaTeam used to call the method to the parameter NcaaTeam’s schedule ArrayList
addScore – 3 pts	
+1	Student correctly names method with correct parameters and objects returned
+1	Student correctly adds new Score object to NcaaTeam object calling the method’s results ArrayList (+1/2 for construction of new Score Object, +1/2 for correctly adding to results)
+1	Student correctly adds a copy of the Score object previously created to the parameter NcaaTeam’s roster
computeWin Method – 5 pts	
+1	Student correctly names method with correct parameters and objects returned. Student also declares all necessary doubles in method to count wins and losses (or total games)
+1	Student is able to correctly call the team and score correctly from the Score class.
+1	Student creates a for loop that correctly goes through all score records in roster without an out of bounds exception error
+1	Student correctly creates conditional logic to compute a win or loss of the team. (.equals() + 1)
+1	Student correctly stores double value representing winRatio correctly for the NcaaTeam object calling the method and returns that value
15	Total points

Potential Solution:

```java
import java.util.*;
class NcaaTeam{
  private String name;
  private ArrayList<NcaaTeam> schedule;
  private ArrayList<Score> results;
  private double winRatio;

  public NcaaTeam(String name){
    this.name = name;
    this.schedule = new ArrayList<NcaaTeam>();
    this.results = new ArrayList<Score>();
    this.winRatio = 0.0;

  }

  public void scheduleAdd(NcaaTeam team)
  {
    this.schedule.add(team);
    team.schedule.add(this);
  }

  public void addScore(NcaaTeam team, int num1, int num2)
  {
    this.results.add(new Score(this, team, num1, num2));
    team.results.add(this.results.get(this.results.size()-1));
  }

  public double computeWin(){
    double win = 0;
    double loss = 0;
    for(Score a:  results)
    {
      if(a.getScore(1)>a.getScore(2))
      {
        if(a.obtainTeam(1).equals(this)) win++;
        else loss++;
      }
      if(a.getScore(1)<a.getScore(2))
      {
        if(a.obtainTeam(1).equals(this)) loss++;
        else win++;
      }
    }
    winRatio = win/(win+loss);
    return winRatio;
  }
}
```