CSC 226 FALL 2023
ALGORITHMS AND DATA STRUCTURES II
ASSIGNMENT 6 - PROGRAM
UNIVERSITY OF VICTORIA

1 Programming Assignment

The assignment is to implement an algorithm to determine which teams in a sports league division have been eliminated from the playoffs. A Java template has been provided containing an empty method BaseballElimination, which takes a single argument consisting of a Scanner object which connects you to the input consisting of the data related to a division of teams within a sports league. The expected behavior of the method is as follows:

Input: A baseball division consisting of n teams.
Output: An ArrayList consisting of all the teams that have been eliminated from the league playoffs.

We will assume that the team with the most wins in the division at the end of the season is the only team to make the playoffs. So, a team is eliminated from the playoffs if they cannot possibly win as many games as any other team in the division.

You may use the FordFulkerson, FlowNetwork, and FlowEdge classes from the Algorithms (Sedgewick) textbook through the algs4.jar file. You may not change any of these classes and thus you do not need to include this code in your program.

I have included a pdf of section 16.4 (two pages) from the Algorithm Design and Applications textbook. You can use this to determine how the reduction algorithm works. To get full marks in this assignment, you need to figure out how to reduce this problem to a network flow problem in order to solve it.

2 Input Format

The testing code in the main function of the template reads an input file containing all the necessary data for a sports division, relevant to this problem. It passes this input file as a Scanner stream to the BaseballElimination method which builds an ArrayList called eliminated, a list of all the eliminated teams you have found.

The input files will have the following format for a division in a league: an integer representing the number of teams in the division followed by, for each team, the team name, their number of wins, their number of remaining games (total), an ordered list of the number of remaining inter-divisional games between the team and the rest of the division (one for each divisional team). Thus, for the following example,

i	Team	wins	games left	vs. 0	vs. 1	vs. 2	vs. 3	vs. 4
0	New York	75	28	-	3	8	7	3
1	Baltimore	71	28	3	-	2	7	4
2	Boston	69	27	8	2	-	0	0
3	Toronto	63	27	7	7	0	-	0
4	Detroit	49	27	3	4	0	0	-

the input file, teams5.txt, is

```
5
New_York 75 28 0 3 8 7 3
Baltimore 71 28 3 0 2 7 4
Boston 69 27 8 2 0 0 0
Toronto 63 27 7 7 0 0 0
Detroit 49 27 3 4 0 0 0
```

3 Test Datasets

A collection of test input files have been included in in a zip file. Some contain historical data from real sports divisions and some are fictional. Not all of them are actually baseball. I have also included the output of my code for each file. You will notice that my output contains more information than I am asking from you, specifically I am printing the state of the maxflow in each case. Your code will only print the final output, a list of eliminated teams.

4 Sample Run

The output of a model solution on the division above is given in the listing below.

```
Reading input values from teams5.txt.
Teams eliminated: [Detroit]
```

5 Evaluation Criteria

The programming assignment will be marked out of 20, based on a combination of automated testing and human inspection, based on the criteria in the table below.

Score (/25)	Description
0 - 5	Submission does not compile or does not conform to the provided template.
5 - 10	The program compiles but reports the eliminated teams incorrectly or does not use a maxflow algorithm to solve the problem.
10 - 20	The implemented algorithm is accurate on all tested inputs and uses maxflow. (Note, there is a large range here. More efficient code will be rewarded).

To be properly tested, every submission must compile correctly as submitted, and must be based on the provided template. You may only submit one source file. **If your submission does not compile for any reason (even trivial mistakes like typos), or was not based on the template, it will receive at most 5 out of 20**. The best way to make sure your submission is correct is to download it from Brightspace after submitting and test it. You are not permitted to revise your submission after the due date, and late submissions will not be accepted, so you should ensure that you have submitted the correct version of your code before the due date.