Programming II (Java)
Term 1, 2018-19
Lab 2

We would like to simulate a knock-out style tournament such as Wimbledon or the final rounds of the World Cup. The Tournament starts with *n* Teams (n is a power of 2), each with a unique name.

The teams are arranged in some order (say, a random draw). The tournament proceeds in Rounds, where in each round, Matches are played by pairs of teams (based on the original draw). The winner of each match advances to the next round, where again successive pairs of teams (winners from the previous round) play matches, using the same ordering as the original draw.

The winner of each match is determined by a relative ordering (or ranking) of each pair of teams. The matrix below describes the outcome for each such pair: a "-1" implies the team specified by the row wins, and a "1" implies the team specified by the column wins.

For example, if there are 4 teams A, B, C, and D, and the relative rating matrix is as below:

Teams	A	B	C	D
A	0	-1	1	1
B	1	0	-1	1
C	-1	1	0	-1
D	-1	-1	1	0

This implies that A will always beat B, while D will always beat A and B, but lose to C.

Hence, if the draw order is:
C B A D

The output should be:
Round 1 Winners: B D
Round 2 Winners: D

Write a Java program to model this, and to run through the matches and identify the winner of the tournament. We would like the results of each match and round to be retained for subsequent analysis. The program should print out the results of each match, round by round, **after** the completion of the tournament. Assume the above scenario is the sample input data. The program will be tested on other inputs too.

You can assume that n is not very large - say, a max of 1024.

You should have the following classes to represent Tournament, Team, Match.
The main should only instantiate Tournament and Team and any other objects and invoke a method on Tournament that will, in turn, play the matches.

Note that the code for determination of the winner can be implemented with either recursion or iteration. Try both approaches.

Variant (optional):
Assume that we want to try out many different rules for deciding the winner of a given match. Not all of these may be known when the program is first designed. However, you would like to not have to change too much of the code every time a new rule is implemented. How would you change the design of classes to allow this kind of “scalability”? Is it possible to design the code such that there are no changes to any of the existing classes except the “main” method?