   
   The *veto rule* is the following social choice rule:
   
   - Every agent names his least desired outcome.
   - The rule then selects the outcome which was named the least number of times.
   - In the case of ties, assume for the purpose of our analysis that all least-named outcomes are returned.

   (a) [5 Points] Formalize the veto rule as a *positional scoring rule*.
   
   (b) [5 Points] Show that it is not *Condorcet consistent* by giving a counterexample.


   (a) [5 Points] Prove that the Kemeny rule is equivalent to the majority rule if there are only two outcomes (but arbitrarily many agents).

   (b) [20 Points] Implement the Kemeny rule as an integer program using JOpt. A code skeleton is provided in the class `kemenyRule.java`. The IP formulation can be found in section 15.1.4 in the readings.

   You can use the methods `getKemenyScore` and `getResultingScore` to test the result. Do not change the signatures of any existing methods or members. (it’s OK to add new ones, though)

   (c) [5 Points] Prove that the Kemeny rule cannot be represented as a *positional scoring rule*. 


(d) [5 Points] Show that the Kemeny rule does not satisfy independence of clones. To do this, add two clones of outcome c in the following example and compute all possible results of the Kemeny rule with and without the clones.

\[4 \circ c \succ a \succ b \quad 2 \circ c \succ b \succ a \quad 3 \circ c \succ b \succ a \quad 1 \circ c \succ b \succ a\]

(Hint: For computing the Kemeny result, a triple of clones can be treated as a single outcome, with three times the weight in regard to wins and defeats.)

(e) [5 Points] Show the following strong version of the continuity property:

When the set of voters is replicated any finite number of times, the results of the Kemeny rule stay the same. (In case of ties, we consider the set of all possible results)

(Hint: What is the effect of the replication on the weighted majority graph? What is then the effect on the Kemeny scores as defined in equation (15.1)?)

3. [Bonus Assignment] Implement the Borda positional scoring rule and investigate the runtime of your code vs. the Kemeny rule on randomly generated preference profiles:

Make a 3D plot of the runtimes of the two rules against the number \(n\) of agents and \(m\) of outcomes. Find a suitable range of \(n\) and \(m\) where one can see a difference in the runtime behaviors.

Explain your findings.