Programming Assignment Checklist: Baseball Elimination

Frequently Asked Questions

How do I read in the data? We recommend using the `readInt()` and `readString()` methods from In.java.

How efficient should my program be? You should be able to handle all of the test input files provided (say, in less than a minute). Do not worry about over-optimizing your program because the data sets that arise in real applications are tiny.

What should I return if there is more than one certificate of elimination? Return any such subset.

Must I output the teams in the same order as in the input file (as does the reference solution)? No, any order is fine.

Should `certificateOfElimination()` work even if `isEliminated()` has not been called by the client first? Absolutely. It is bad design (and a violation of the API) for the success of calling one method to depend on the client previously calling another method.

How do I compute the mincut? The `inCut()` method in FordFulkerson.java takes a vertex as an argument and returns true if that vertex is on the *s*-side of the mincut.

How do I specify an infinite capacity for an edge? Use `Double.POSITIVE_INFINITY`.

What would cause `FordFulkerson.java` to throw a runtime exception with the message `"Edge does not satisfy capacity constraints: ..."`? Did you create an edge with negative capacity?

My program runs much faster in practice than my theoretical analysis guarantees. Should I be concerned? No, there are a number of reasons why your program will perform better than your worst-case guarantee.

- If a team is eliminated for a trivial reason, your code may run much faster because it avoids computing a maxflow.
- If there are no games between most pairs of teams, then the flow network you construct may have many fewer edges than in the worst case.
- The flow networks that arise in the baseball elimination problem have special structure, e.g., they are bipartite and the edge capacities are small integers. As a result, the Ford-Fulkerson algorithm performs significantly faster than its worst-case guarantee of $V E^2$.

Input

Input and testing. The directory baseball contains some sample input files. For convenience, baseball-testing.zip contains all of these files bundled together.

Testing. For reference, the teams below are mathematically eliminated for *nontrivial* reasons. By nontrivial, we mean that the certificate of elimination requires two or more teams. If a team is trivially eliminated, it does not appear in the list below.

- teams4.txt: Philadelphia.
- teams4a.txt: Ghaddafi.
- teams5.txt: Detroit.

- teams7.txt: Ireland.
- teams24.txt: Team13.
- teams32.txt: Team25, Team29.
- teams36.txt: Team21.
- teams42.txt: Team6, Team15, Team25.
- teams48.txt: Team6, Team23, Team47.
- teams54.txt: Team3, Team29, Team37, Team50.

To verify that you are returning a valid certificate of elimination R, compute $a(R) = (w(R) + g(R)) / |R|$, where $w(R)$ is the total number of wins of teams in R, $g(R)$ is the total number of remaining games between teams in R, and $|R|$ is the number of teams in R. Check that $a(R)$ is greater than the maximum number of games the eliminated team can win

Possible Progress Steps

These are purely suggestions for how you might make progress. You do not have to follow these steps.

- Write code to read in the input file and store the data.
- Draw by hand the `FlowNetwork` graph that you want to construct for a few small examples. Write the code to construct the `FlowNetwork`, print it out using the `toString()` method, and and make sure that it matches your intent. Do not continue until you have thoroughly tested this stage.
- Compute the maxflow and mincut using the `FordFulkerson` data type. You can access the value of the flow with the `value()` method; you can identify which vertices are on the source side of the mincut with the `inCut()` method.
- The `BaseballElimination` API contains the public methods that you will implement. For modularity, you will want to add some private helper methods of your own.

Your program shouldn't be too long—ours is less than 200 lines. If things get complicated, take a step back and re-think your approach.

Enrichment links

- A group of researchers at Berkeley maintain a website where you can view the baseball eliminationnumbers as the season unfolds.