Algorithms

- Greedy by performance rate:
  A very naïve way to get a solution would be always choosing the neighbor with the highest performance rate.
  1. Starting from an arbitrary vertex v in the graph, for all its neighbors, choose vertex u with highest performance rate and add it to the team and remove the previous vertex v from the graph. Set u as the current vertex and repeat the same procedure until reaching a sink vertex. A team is found
  2. Keep doing 1 until there is no vertex in the graph
  3. Return all the teams

  This algorithm is fast since it only takes linear time so it is used for large graphs when all the other methods fail. It is implemented as the greedyByWeight() method in the class Horse.
- Find the "longest" team:
  Since the overall score is also related to the length of each team, a longer team would tend to score higher. So we can also greedily find the longest teams in the graph. However, the longest path problem is NP-complete itself, we can use some heuristic method. According to some literature, a good approximation for a sparse graph would be the following:
  1. Starting from an arbitrary vertex v in the graph, for each of its neighbors u, run DFS from u and record the size of the subgraph reachable from u. Choose the neighbor with the largest potentially reachable subgraph as the successor of v and add it to the team. Repeat the same procedure until reaching a sink vertex. A team is found
  2. Keep doing 1 until there is no vertex in the graph
  3. Return all the teams

  This algorithm takes polynomial time and gets stuck when the size of the graph is too large or the graph is too dense, but it does perform better than the naïve one in most cases. So I set a threshold of the complexity of the graph under which this algorithm would be run. It is implemented as the greedyByLength() method in the class Horse.
- Find team members with potentially higher overall scores:
  This is basically the same algorithm as the one above except for that when choosing a neighbor of v, we choose the one with the highest potentially achievable score of the reachable subgraph. It is implemented as the greedyByScore() method in the class Horse.

Reference: Approximating Longest Path, Andreas Björklund