1 Main Idea

In this problem, there are two types of professional wrestlers: babyfaces ("good guys") and heels ("bad guys"), and between any two wrestlers, there may or may not be a rivalry. Given n professional wrestlers and a list of r pairs of wrestlers with rivalries, we want to determine whether it is possible to designate some of the wrestlers as babyfaces and the remainder as heels such that each rivalry is between a babyface and a heel in $O(n + r)$ time.

The main idea of the solution is to represent the wrestlers as vertices in a graph, and the rivalries as edges between the vertices. Then, we use DFS to traverse the graph and assign babyface and heel labels to the vertices. In the end, if we can assign labels to all the vertices, then we return the labels; otherwise, we return "No".

2 Pseudocode

In this pseudocode, we represent the wrestlers as numbers from 1 to n. Arrays are indexed starting from 1. The graph G is represented as an adjacency list, where $G[i]$ is the list of vertices adjacent to vertex i. We assume that r is a list of pairs of wrestlers with rivalries, where each pair is represented as a tuple (u, v). We have an array of labels $labels$ where the index i corresponds to the label of wrestler i.

Algorithm 1: Assign Labels to Wrestlers

Input: A list of n professional wrestlers and r pairs of wrestlers with rivalries

Output: A list of labels for the wrestlers or "No" if no such assignment exists

```
G = [[] for i = 1 to n] ;                          // Adjacency list
labels = ["" for i = 1 to n] ;                     // Array of labels
for (w_1, w_2) in r do
    G[w_1].append(w_2);
    G[w_2].append(w_1);
end
for i = 1 to n do
    if labels[i] = "" then
        if not DFS(G, labels, i, "babyface") then
            return "No";
        end
    end
end
babyfaces = [i for i = 1 to n if labels[i] = "babyface"];
heels = [i for i = 1 to n if labels[i] = "heel"];
return babyfaces, heels;
```

Algorithm 2: DFS

Input: Reference to the graph G, reference to the array of labels *labels*, wrestler w, label l

Output: True if the labels can be assigned, False otherwise

```
labels[w] = l;
for rival in G[w] do
    if labels[rival] = l then
        return False;
    end
    if labels[rival] = "" then
        opposite = "babyface" if l = "heel" else "heel";
        if not DFS(G, labels, rival, opposite) then
            return False;
        end
    end
    return True;
end
```

3 Proof of Correctness

4 Time Complexity Analysis