CS 325 - Activity 6

You may work in groups with up to 3 students. When submitting solutions in Gradescope select a page for each problem and the students in your group.

Written: (5 pts)

The Wrestler Problem: Suppose there are two types of professional wrestlers: “Babyfaces” (“good guys”) and “Heels” (“bad guys”). Between any pair of professional wrestlers, there may or may not be a rivalry. Suppose we have n wrestlers and we have a list of m pairs of rivalries.

a) Give a written description of an algorithm that determines 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. If it is possible output Possible otherwise output Impossible.

Have a graph of the wrestlers, with the edges determining a rivalry (ex babyface = blue, heels = red). We need to check if the whole graph is bipartite, determine if it’s possible or not, and output that.

b) Give pseudocode for your algorithm:

```
1. Let V be the size of the graph.
2. Create an array of size V containing the wrestlers, with 1 representing Babyfaces and 0
   representing Heels. Let this array be called Arr.
3. Assign -1 in array from indices 0 to V.
4. Arbitrarily assign value of 1 (Babyface) to the source, which in this case starts at index 0 (Heel)
   for the adjacency matrix.
5. Queue the source.
6. While the queue is not empty:
   a. Dequeue a vertex.
   b. For each adjacent vertex.
      i. If an edge exists, assign the inverse value.
      ii. Add to the queue.
      iii. If an edge exists already with the same value, return False.
7. Return True.
```

c) What is the running time of your algorithm?

$O(n^2)\ or\ O(n + m)$

Code: (10 pts)
Implement your algorithm for Babyfaces vs Heels in C++.

Input: The input contains the number of wrestlers, n (1,..,n) , followed by the number of rivalries m and rivalries listed in pairs, x y where 1 ≤ x, y ≤ n and x≠y.

Output: Results are outputted to the terminal.

- Possible or Impossible

Sample:

Input:

```
4
4
1 2
1 3
4 2
4 3
```

Output:

```
Possible
```

You can use the code template provided. The name of file you submit to Gradescope must be **act6.cpp**. You may submit multiple times. Select all group member names each time you submit and also include the names of the group member in your comments.