Riverside Games - Alan Yan

Observe that this is just a rooted tree and we are travelling down the branches until we hit a leaf. We define two functions $A, B : V \to \{True, False\}$. We define $A(v) = Keith\ turn\ on\ v,\ Keith\ wins$ and $B(v) = Ben\ turn\ on\ v,\ Keith\ wins$. Then, we have the following two equations

$$A(v) = \bigcup_{w \in \Gamma(v)} B(w)$$

$$B(v) = \bigcap_{w \in \Gamma(v)} A(w)$$

The rest is dynamic programming. My code is given below.

```
import java.util.*;
import java.io.*;
public class GraphGame {
    static String[] lines;
    ArrayList<Integer>[] tree;
    int n;
    boolean[] memoA, memoB; //not needed but wtv
    boolean[] a, b;
    GraphGame(ArrayList<Integer>[] tree) {
       this.tree = tree;
       n = tree.length;
       memoA = new boolean[n];
       memoB = new boolean[n];
       a = new boolean[n];
       b = new boolean[n];
    }

    boolean solve() {
       return A(0);
    }
    //Keith wins, his move
    boolean A(int src) {
       if(memoA[src])
             return a[src];
       memoA[src] = true;
       if(tree[src].size() == 0) {
             a[src] = false;
```

```
            return a[src];
        }
        for(int v : tree[src]) {
            if(B(v)) {
                a[src] = true;
                return true;
            }
        }
        a[src] = false;
        return false;
    }

    //Keith wins, Ben's move
    boolean B(int src) {
        if(memoB[src])
            return b[src];
        memoB[src] = true;
        if(tree[src].size() == 0) {
            b[src] = true;
            return true;
        }
        for(int v : tree[src]) {
            if(!A(v)) {
                b[src] = false;
                return false;
            }
        }
        b[src] = true;
        return true;
    }
    public static void main(String[] args) throws IOException {
        System.out.print(SOLVE());
    }
    public static void read() throws IOException {
        BufferedReader br = new BufferedReader(new FileReader("file.txt"));
        lines = new String[100000];
        for(int i = 0; i < lines.length; i++)
            lines[i] = br.readLine();
        br.close();
    }

    public static int SOLVE() throws IOException{
```

```
        read();
        int count = 0;
        for(int i = 0; i < lines.length; i++) {
              String[] set = lines[i].split(" ");
              ArrayList<Integer>[] tree = new ArrayList[100];
              for(int j = 0; j < 100; j++)
                    tree[j] = new ArrayList<Integer>();
              for(int k = 0; k < 99; k++)

tree[Integer.parseInt(set[2*k])].add(Integer.parseInt(set[2*k+1]));
              GraphGame test = new GraphGame(tree);
              if(test.solve()) count++;
        }
        return count;
    }
}
```