As you can see the value of **n** is small. It also can be solved by bitmask DP. Let assume arr contains players favorite number and **dp[mask, i]** is true if we can find a way to assign 1 to n to each favorite number from this state otherwise where i represent the i-th player and mask represent the number of values from 1 to n are taken.

dp [mask, i] |= ∀j ∈ dp [mask with jth bit on, i+1]; where j = {x | x ∈ [1,n], x-th bit in mask is off and arr[i]%x = 0}dp[mask with all 1st to nth bit on] = true

To handle the lexicographically smallest one you have to find the smallest j for dp [mask, i] where dp [mask with jth bit on, i+1] is true.

It can also solved by Bipartite Matching. Construct a graph where favorite number are connected to their divisor number between 1 to n. Then apply BPM on that graph. If you find number of matching is less then n then the answer would be “Again Runners-up” otherwise you can print the matched number. As we want lexicographically smallest one, you have to modify the BPM to handle this.