# Presentation of Solutions

ACM ICPC - Tehran Site Qualifications Round 1

Isfahan University of Technology ACM Student Chapter,

November 2019

Created by: Alireza Omidi & Mehran Aghabozorgi

## Outline

## A. Another Story of Moein-Ali

Check whether input sequence is correctly sorted in ascending order.

- Solution 1
  - Check whether $x_i \leq x_{i+1}$ for all i.
  - $\Rightarrow O(n)$
- Solution 2
  - Don't think, just calculate the extreme property, i.e. calculate all $x_{i,j}$ and print "no" if any of those is less than 0, otherwise print "yes".
  - $\Rightarrow O(n^2)$ - fast enough!

## B. Shuffle!

- Count how many different songs are in the intervals of length $s$.
- Update for a next interval in $O(1)$ time by adding and removing one song
- Then check which positions are valid

## C. World cup

- Find strongly connected components with Tarjan's algorithm or any of your choice!
- Consider DAG of strongly connected components
- If this DAG has exactly one source (SCC with indegree zero), print out the nodes of this SCC
- if not, Marc made a fault
- Most common error: Didn't use SCCs, tried $n$ dfs instead.

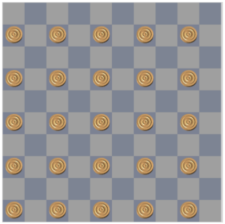
## D. Draughts

**Problem**: you are given a $10 \times 10$ draughts board. How many captures can the white player perform in his next move?

- simply use backtracking to check all possible moves and hope it runs fast enough...
- ...or prove it.

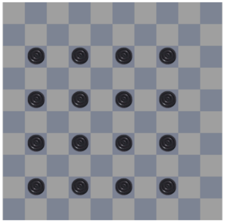
## D. Draughts

When you start from one of these fields you can never visit any other field.



## D. Draughts

So you can only capture these pieces – there are 16 of them.



In every step (apart from the first one) you can go in one of three directions (you cannot go back) so the number of moves to check is at most:

$$3^{16} \approx 40\ 000\ 000$$

## E. Isfahan Crisis

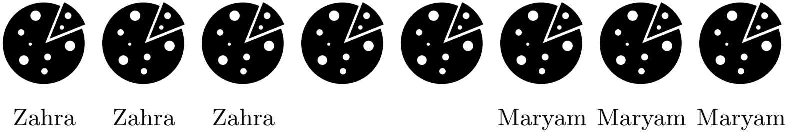
**Problem**: Given various statements about the relative wealth of the Isfahanies, decide whether there is a contradiction in their statements.

- Read input as directed graph with an edge from the poorer to the richer Isfahani.
- Check if the graph is cycle-free (e.g. with DFS).

## F. Rick vs. Morty

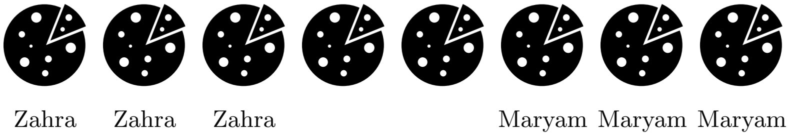
- Make a bipartite graph with Rick lovers and Morty lovers as vertices
- Add an edge if their votes are incompatible
- Problem now is: find minimum vertex cover
- Equivalent to maximum matching for bipartite graphs

## G. Pizza Voting



- Maryam will veto the last third
- Zahra will veto the first third
- You can choose any in the middle third
- Rounding at the border of thirds

## G. Pizza Voting



- Maryam will veto the last third
- Zahra will veto the first third
- You can choose any in the middle third
- Rounding at the border of thirds
- `puts((i > n/3 && i <= n-(n+1)/3) ?  "YES" :  "NO");`

## H. Equilibrium Mobile

- All weights at a certain level must have the same weight
- All weights one level higher must have twice that weight, and so on
- Calculate all $2^{\text{depth}} \times$ weight
- Use 64-bit integers for that
- Find the most recurring one (e.g. by sorting first)

## I. Equator

Before visiting each city we may either:

- Continue visiting the cities ahead, if it is profitable, or
- Get out of the planet if it is not

So:

$$p(i) = \begin{cases} 0, & \text{if } i < 0 \\ c_i + \max(0, p(i-1)), & \text{otherwise} \end{cases}$$

where $p(i)$ is the maximum amount of money we get if we start robbing from city $i$ on the equator.

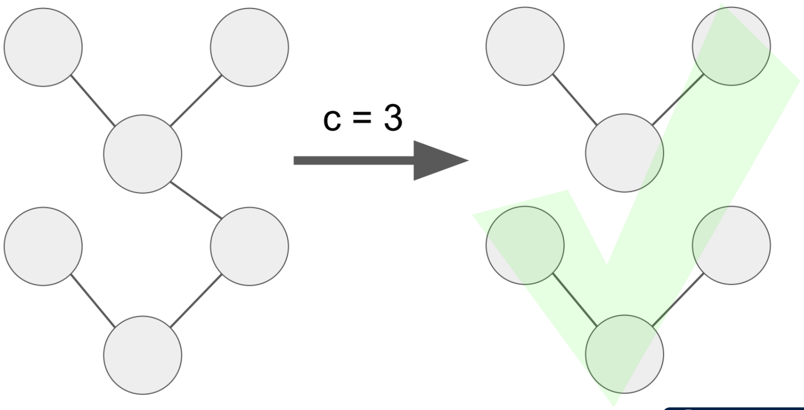
- The final answer is $max_{-1 \leq i < n} p(i)$.

## J. No One Loves Sheikhbahaei!

- Naive solution requires $C \times O \approx 10^{11}$ operations
- Coord compression reduces $C$ to $4 \times O$, $10^{10}$ ops
- $\Rightarrow$ Advanced data structure necessary
- Segment tree with lazy propagation
- Inner nodes represent intervals
- Store minimum, maximum and value in each node
- Propagate values only when necessary
- Answer queries in $O(logn)$
- (Even more efficient: combine both ideas but not necessary)

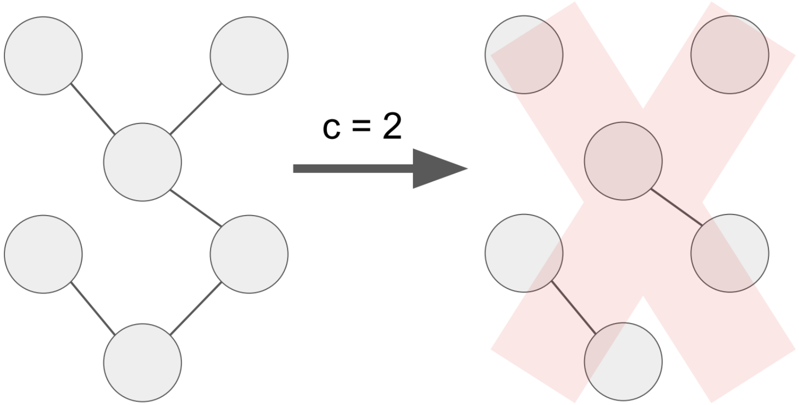
## K. Justified Jungle

Problem: Given a tree, find all integers $c$, such that we can cut a tree into components of size $c$.



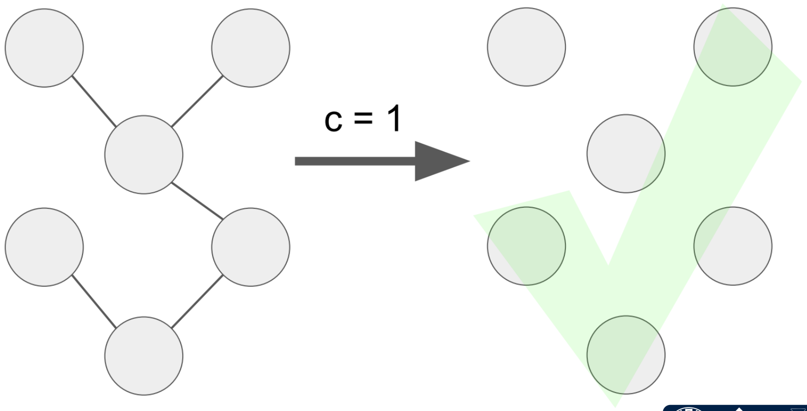
## K. Justified Jungle

Problem: Given a tree, find all integers $c$, such that we can cut a tree into components of size $c$.



## K. Justified Jungle

Problem: Given a tree, find all integers $c$, such that we can cut a tree into components of size $c$.



## K. Justified Jungle

- The tree size needs to be divisible by $c$.
- There aren't that many divisors: worst case 240 for $n = 720720$.
- We can try each divisor separately.

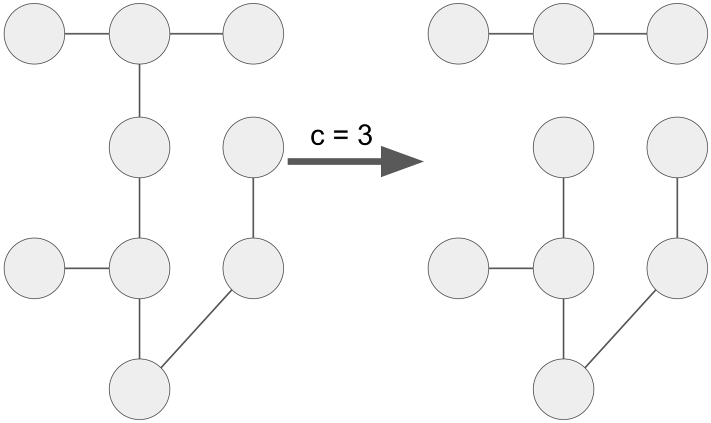
## K. Justified Jungle

Problem reduces: Given a tree of size $n$ and integer $c$, such that $c|n$, can we cut it into components of size $c$?
Iterative algorithm:

- If $n = c$: done.
- Otherwise:
    - Find an edge that divides the tree into subtrees of sizes $c$ and $n - c$.
    - If there is no such edge: impossible.
    - Otherwise: Cut the edge and repeat the algorithm on the subtree of size $n - c$.

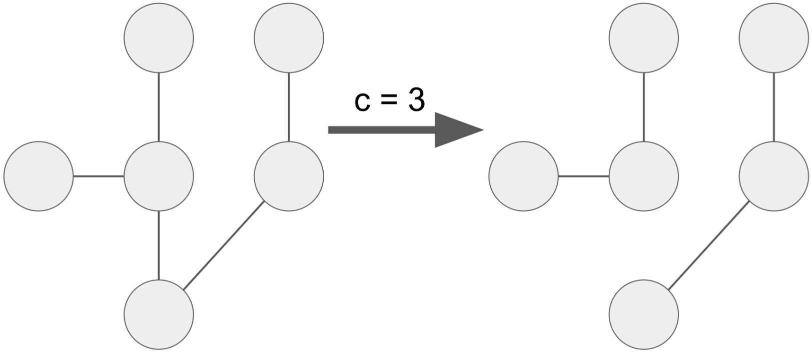
# K. Justified Jungle

Problem: Given a tree of size $n$ and integer $c$, such that $c|n$, can we cut it into components of size $c$?



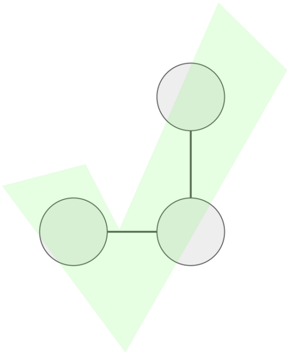
## K. Justified Jungle

Problem: Given a tree of size $n$ and integer $c$, such that $c|n$, can we cut it into components of size $c$?



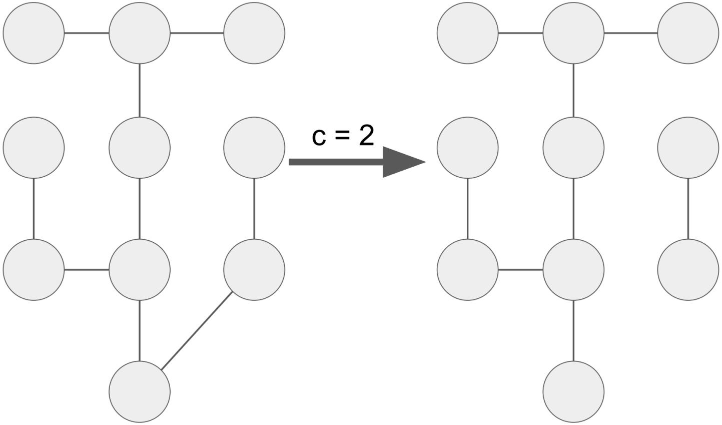
# K. Justified Jungle

Problem: Given a tree of size $n$ and integer $c$, such that $c|n$, can we cut it into components of size $c$?



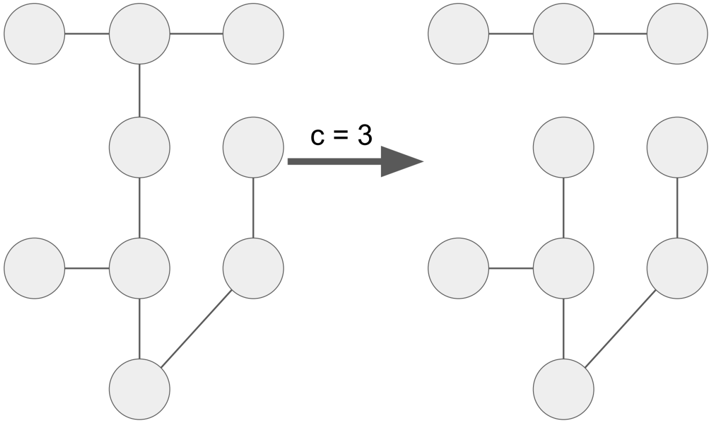
# K. Justified Jungle

Problem: Given a tree of size $n$ and integer $c$, such that $c|n$, can we cut it into components of size $c$?



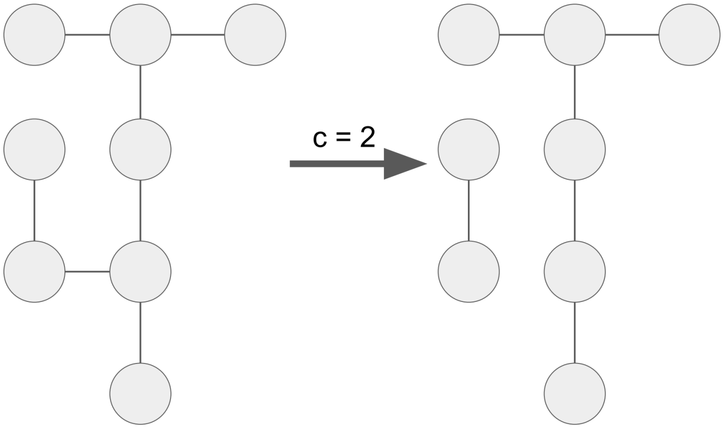
## K. Justified Jungle

Problem: Given a tree of size $n$ and integer $c$, such that $c|n$, can we cut it into components of size $c$?



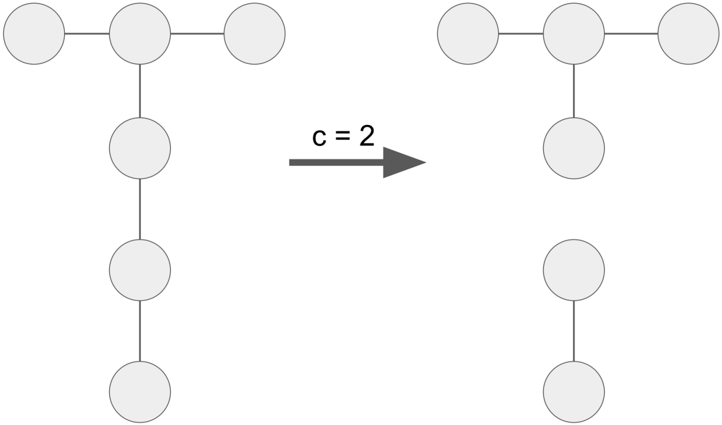
## K. Justified Jungle

Problem: Given a tree of size $n$ and integer $c$, such that $c|n$, can we cut it into components of size $c$?



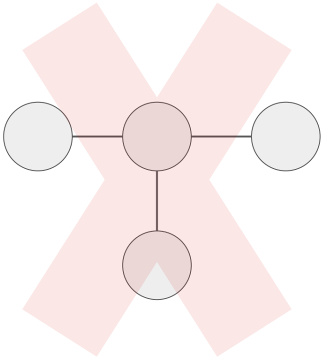
## K. Justified Jungle

Problem: Given a tree of size $n$ and integer $c$, such that $c|n$, can we cut it into components of size $c$?



## K. Justified Jungle

Problem: Given a tree of size $n$ and integer $c$, such that $c|n$, can we cut it into components of size $c$?

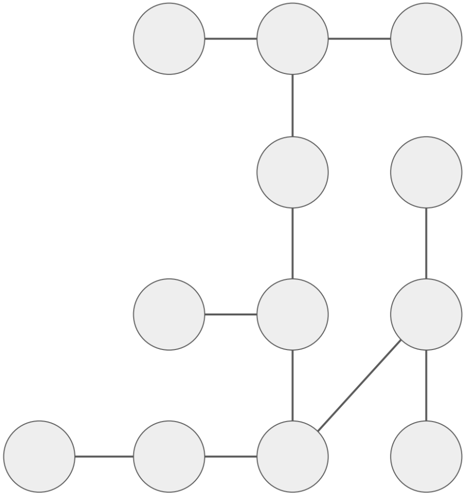


## K. Justified Jungle

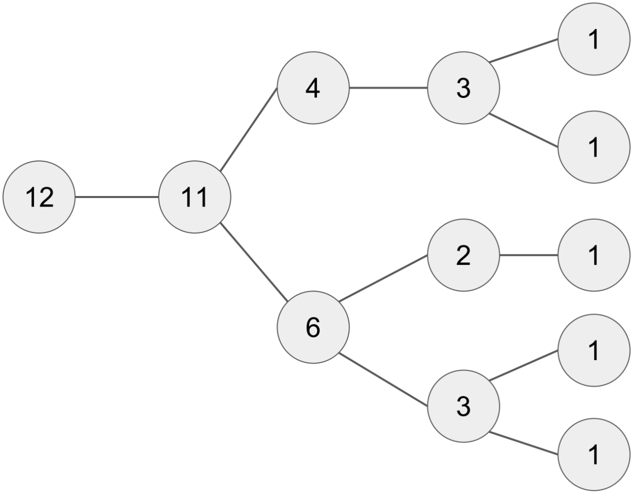
Problem: Given a tree of size $n$ and integer $c$, such that $c|n$, can we cut it into components of size $c$?
Iterative algorithm is difficult to implement in O(n), and might time out.
Simplified algorithm:

- Root the tree and compute the size of each subtree (only once, no need to repeat for each divisor).
- Find edges with subtrees sizes equal to a multiple of $c$. Those are the ones we'll end up cutting.
- If the number of found edges is equal to $\frac{n}{c} - 1$: yes!
- Otherwise: no!

# K. Justified Jungle

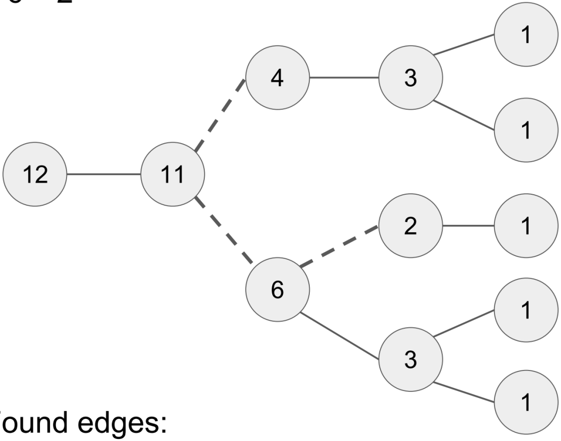


## K. Justified Jungle



# K. Justified Jungle

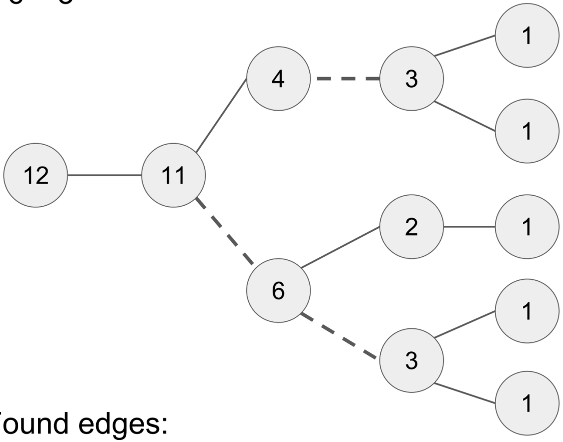
c = 2



Found edges:
3 ≠ n / c − 1 → NO

# K. Justified Jungle

c = 3



Found edges:
3 = n / c − 1 $\rightarrow$ YES