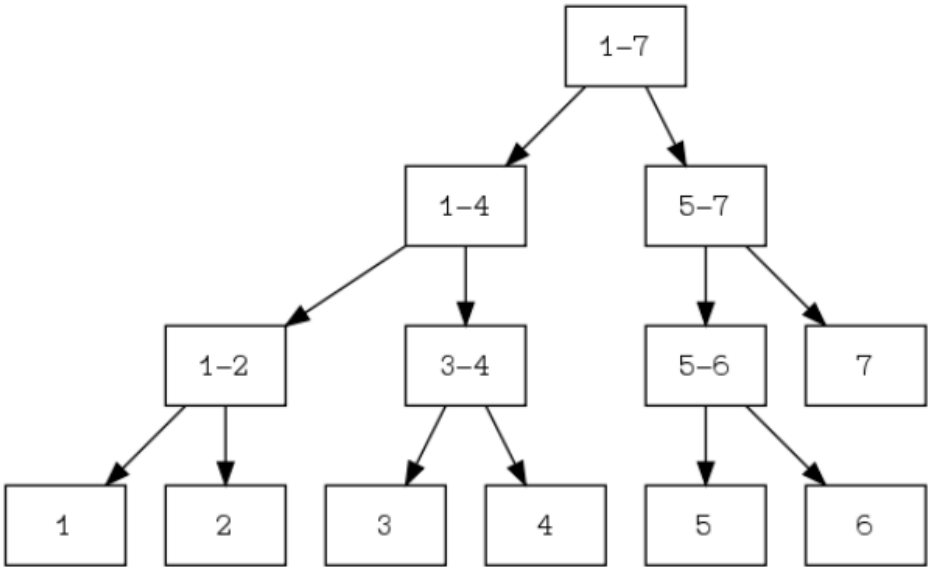
Garment Rush

• Note that since, m, r = 100000, you cannot update the stack in O(m) time.
• Binary Indexed Tree / Fenwick tree does the job in O(log(m)) time.



• At every step, take garment x and put it back on -1, -2, -3, ...

Candy Distribution

It's a pretty straight forward dp solution. Brute force for this would be 2^n possible states. That would definitely time out. The idea is that if you realise all the game states (left,right) would be at max n^2, it can be solved with a n^2 dp easily. Let me try explaining with a pseudo code,

```
dp[first][last]
    ar[n] // stores the number of candies in each packet.
    int go(int first, int last) {
        if (dp[first][last]!=-1) return dp[first][last]; // if state already computed
        else if (first==last) return ar[first]; // base case
        return dp[first][last]=max(ar[first]-go(first+1,last),ar[last]-go(first,last-1));
    }
```

Finally dp[0][n-1] would give me the answer.

Circles of Doom

This problem was pretty straight forward.
Each roller's answer will be r[i]/r[0]

P versus NP

To solve this problem, consider each person as a node of a graph. Let's add two more nodes to this graph called "source" and "sink". Now, make an undirected edge between two people if they are collaborators and let the edge weight be one. Also, if a person believes in P=NP, then make an undirected edge from "source" to that person of edge weight one. Similarly, if the person believes in P!=NP, then make an undirected edge from "sink" to that person of weight one. Now, if you observe the structure of the graph carefully, you'll notice that the answer to the original problem is the min-cut of the graph (http://en.wikipedia.org/wiki/Minimum_cut). The detailed explanation of this observation is left as an exercise to the reader (Try a few small cases by hand and you'll see why this is true). This can be found by finding the maximum flow through the graph (http://en.wikipedia.org/wiki/Max-flow_min-cut_theorem) where the weight of an edge acts as its capacity. Hence you can use any standard max-flow algorithm (such as Ford–Fulkerson) to obtain the answer.

Lazy Bob

Key Points:

1. Given the description of the N recipes, we can sort them first by decreasing order of Alice's time and if for two recipes, Alice's time is equal, then sort them in increasing order by Bob's time. This ordering of the recipes will give us the order in which Bob will select the recipes. Let's call this ordering G. So, given any P recipes, Bob will choose them in the order they occur in G.

2. Now, we know the order in which Bob will select the recipes. It should be observed that Bob selects K out of the P recipes given to him by Alice. So, the last (P - K) elements in G will never be selected, no matter which P recipes Alice gives Bob, Bob will choose K out of those P based on the ordering in G, so the last (P-K) recipes in G, will never be chosen, no matter what.

3. Till now, we have established that the last (P-K) elements of G will never be selected by Bob. So, let's focus on the remaining recipes in G. Now let's sort the remaining elements in G first in decreasing order of Bob's time and if for two recipes, Bob's time is equal, then sort them in decreasing order of Alice's time. Let's call this ordering H. This ordering of the recipes will give us the order in which Alice wants to select the recipes.

4. Alice will select the first K recipes in H and she wants to ensure that Bob also selects all of these K recipes. So, what she does is, she all of these K recipes to Bob, and for the remaining (P-K) recipes which she has to pass to Bob, she will find out the greatest index of all the recipes she has already given to Bob in G. Let's call this index J. And she will select the remaining (P-K) recipes from G in such a way that their index in G is greater than J

This process will ensure that Bob always selects the first K recipes in H and leaves the rest for Alice.

Space Walk

The problem is a trivial application of Fermat's last theorem and the existence of Pythagorean triples.
Thus the answer was (x>=5)?(2):(1).

Pulkit’s Dilemma

[Coming soon...stay tuned on our fb event page]