Data Structure Design Questions

1 Kartikland: The Happiest Place in the World

You are a ticket ripper at the happiest place in the world, Kartikland. Each family in your land can rip exactly one ticket, no more no less. Additionally, you want to rip the tickets of younger people before older ones, after all, Kartikland is a place for young ins to drag their parents to spend lots of money. Families may attempt to get around this by sending multiple family members with tickets, but it is up to you, the brave ticket ripper to prevent them from getting more than one ripped ticket. You are provided the familial connections of all families in whatever representation you would like. You should be able to rip the tickets of all people in O(N) time where N is the number of people in line to get their tickets ripped.

2 Sweet Tooth

Pretend that you are the owner of a Sweet Company. Your company makes three sweets- popcorn, candy, and ice cream. In addition to this, each Sweet Category has their own sub categories. For example, you can have Vanilla Ice Cream, Chocolate Ice Cream, Banana Ice Cream etc. Each Sweet has a total of N sub categories. You know how much money each sub category will yield.

As an owner trying to make a profit, you want to get lots of money; however, you can only make one item a day, and you only have M days (where M< 3N). Your strategy is to pick the sub category of food that will provide the most money on and then create it that day. You have 2 restrictions though.

- On any 2 consecutive days, you cannot pick items that come from the same Sweet Category. For example, you cannot create an item from Ice Cream one day and the next day pick another item from ice cream. This is enforced until the other 2 Sweet Categories are depleted.
- Once you make a subcategory, you cannot make it again. For example, if you make Banana Ice Cream one day, you can never again make Banana Ice Cream.

Utilize Data Structures that we have discussed earlier to create an algorithm that will allow you to pick the most profit while following the above constraints. You must take no longer than Mlog(N) to pick all the sub categories of Sweets you will make. Hint: You cannot simply put everything inside 1 Priority Queue.

3 Min-nie Mouse and Max-xie Mouse

Use or modify a data structure or set of data structures to do the following operations in the provided time:

- insert: O(log(N))
- getSmallest: O(1)
- getLargest: O(1)
- deleteSmallest: O(log(N))
- deleteLargest: O(log(N))

4 Oh Hi Mark

Imagine you are implementing a basic social media. You want to be able to check if 2 people are friends in constant time. You also want to be able to see if you can see if 2 people have at least 1 mutual friend in O(N) time where N is the total amount of people on the social media. Find an efficient solution to this problem

5 Colise-Yum

You are the rich Roman Emperor Jewelius Caesar. You want to build arenas (possibly more than one) so your subjects can watch gladiator duels and eat wild boars.

- You have V locations where you can build arenas and it costs v_i denarius's to build an arena in the ith location.
- Additionally, any location that does not have an arena must have a path to an arena. You are provided a set of roads, denoted by E, that you can build between any 2 locations i and j, along with their cost denoted by e_{ij}.

Find or modify an algorithm that will minimize the amount of denarius's your kingdom will spend while still fulfilling your constraints.

6 Land Lubber? Sea Lubber? Air Lubber? Just Lubber

You are an avid traveler who wants to travel between the 2 cities in your country which consists of V cities. To go between these cities you have decided that you can narrow down your modes of transportation to dinghy sailing, gliding, and unicycling. For every city in the country, you are given a list of cities that you can get to- this list remains constant regardless of which mode of transportation you pick. The cost to travel between 2 cities i and j via sailing is given as e^s_{ij}, the time for gliding between the same 2 cities is e^g_{ij}, and time for unicycling between these cities is e^u_{ij}. Sadly, you get sea/air/landsick quite frequently so you cannot take the same mode of transportation consecutively- in other words you cannot go into city i with unicycling and leave unicycling. Find an algorithm that computes the fastest route from city i to city j with your constraints. Important note: It may

7 National Bongoola Association

You are the coach of a prestigious Bongoola team. Your job as coach is to make sure that, at any given time, the best players on your team for each positions are on the field. Bongoola, being a great community team sport, invites all people of the community to join. Each player will be assigned a position that they can play- there are a total of M positions- and a unique team number. For this problem, we will use N to be the total amount of people on your team and P to be the people signed up for a certain position.

You have your starting M players, but as the game progresses, you will need to replace them. You will base your replacement of people using the bodacious factor. During the game, only 1 player's bodacious factor goes down, and once they are benched, their bodacious factor doesn't change. Every Z minutes, 1 player on the field will have their bodacious factor decrease by some unknown amount and you will be able to replace the player if needed. Once the next eligible player's (a player who plays the same position) bodacious factor exceeds the current player, they are considered a better choice. You are guaranteed that only 1 of the players at a time will need to be replaced. Finding all the players on your current lineup should take $\Theta(M)$ time, changing a player's bodacious factor should take $\Theta(1)$ time, replacing a player should take $\Theta(log(P))$ time, and putting a new player in the current lineup should take $\Theta(1)$ time.

On the next page, write a detailed description of what data structures you will use to make the operations stated above run in the provided time. State any assumptions you need to regarding anything (no you cannot assume that everything is done by some magical warlock).