1 Design

Nice! You got an interview with Oski Zuckergates, CEO of BearbnBeats, the hot new music streaming startup headquartered in Soda Hall. For each of the following scenarios, determine which data structures (doesn't have to be strictly Java) would give the best performance and what algorithms would be used. Additionally, give the worst-case runtime for any operations listed.

a. BearbnBeats provides users access to millions of songs. Zuckergates has a list of N `(song, album)` pairs. Assuming all album names are unique, find the number of songs in each album.

b. Zuckergates has a list of all N song names in BearbnBeats' database, and wants to query if a given `Song` is in the database. Optimize for both constructing the solution and matching a query.

c. Zuckergates wants to start developing auto-complete for search on BearbnBeats' website. When a user types in the first K characters of a query, we want the website to suggest the number of songs that have the same K character prefix. Assume that no songs have a name longer than M and there are N distinct songs. Optimize for both constructing the solution and matching a query.

d. BearbnBeats runs a subsidiary company, BearBnb, an online marketplace for housing rentals. Bearbnb has a database of N `Trips`. Each `Trip` represents a room reservation and has the following attributes: `Integer customerId, Double reservationCost,` and `Date reservationDate`. Zuckergates, a champion for Big Data, wants to run analytics on Bearbnb's database and query for all `Trips` requested within a range of dates for a given customer. Optimize for both constructing the solution and matching a query.

e. BearbnBeats also handles album deliveries to customers all over the world. BearbnBeats wants to optimize its deliveries, but it only allows each of its trucks to carry one type of album at a time. Therefore, BearbnBeats has the policy to send a truck carrying the album of the earliest uncompleted order, while trying to fulfill as many orders as possible for that album. Each `Order` represents an order for a specific album and has the following attributes: `String albumName, Date orderDate, Integer quantity,` and `Double price`. BearbnBeats must maintain some collection of N `Orders` that optimizes adding new orders and figuring out what products to deliver on its next truck.

2 Weighted Quick Union Trees with Path Compression

Assume we have eight sets, represented by integers 1 through 8, that start off as completely disjoint sets. Draw the WQU Tree after the series of `union()` and `find()` operations with path compression. Write down the result of `find()` operations. Break ties by choosing the smaller integer to be the root.

```
union(2, 3);
union(1, 6);
union(5, 7);
union(8, 4);
union(7, 2);
find(3);
union(6, 4);
union(6, 3);
find(7);
find(8);
```

3 Hash 'Em? I Hardly Knew 'Em!

Consider a `HashSet<String>` object that is implemented using a Hash Table with the following properties. The Hash Table has an initial size of 4, resolves collisions via chaining, and doubles its size immediately once the load factor becomes 1.5. If inserted `String` objects are hashed based on their lengths, then draw out the HashTable after performing the following operations.

```
add("Turkey");
add("Stuffing");
add("Mashed Potatoes");
add("Green Bean");
add("Cranberry");
add("Chicken Pot Pie");
add("Ice Cream");
add("Cats");
```